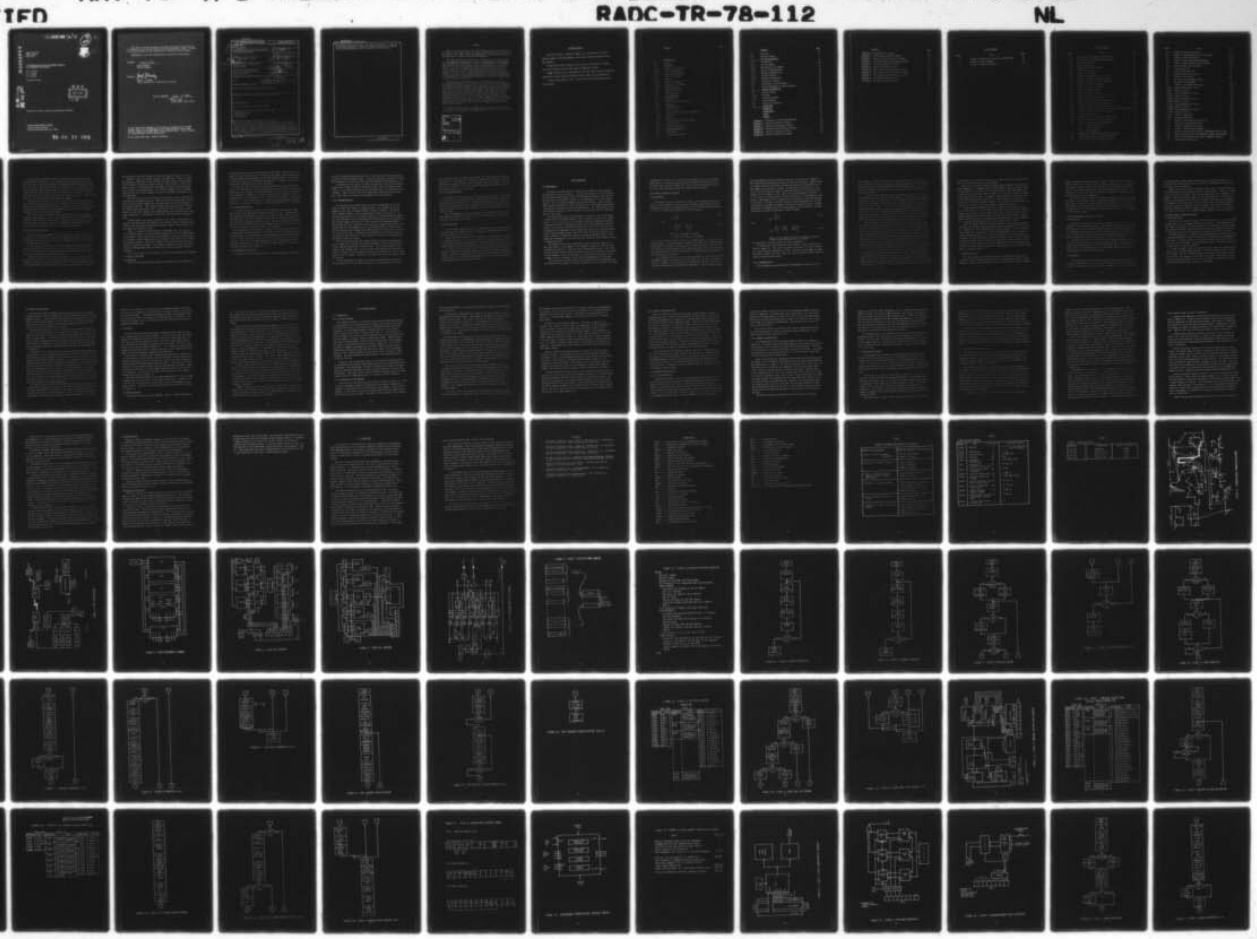
AD-A055 864 PURDUE UNIV LAFAYETTE IND SCHOOL OF ELECTRICAL ENGI--ETC F/G 17/2
A MICROPROCESSOR BASED PERFORMANCE MONITOR FOR COMMUNICATION NE--ETC(U)
MAY 78 M S CALLOW, S C CRIST, B A BLACK F30602-75-C-0082

UNCLASSIFIED RADC-TR-78-112 NL

1 OF 3
AD
A055 864



RADC-TR-78-112
Phase Report
May 1978

A MICROPROCESSOR BASED PERFORMANCE MONITOR FOR COMMUNICATION NETWORKS

M. S. Callow
S. C. Crist
B. A. Black
R. A. Meyer

Clarkson College

ROME AIR DEVELOPMENT CENTER
Air Force Systems Command
Griffiss Air Force Base, New York 13441

This report has been reviewed by the RADC Information Office (OI) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

RADC-TR-78-112 has been reviewed and is approved for publication.

APPROVED: Jacob Scherer

JACOB SCHERER
Project Engineer

APPROVED: Joseph J. Naresky

JOSEPH J. NARESKY
Chief, Reliability & Compatibility Division

FOR THE COMMANDER: John P. Huss

JOHN P. HUSS
Acting Chief, Plans Office

If your address has changed or if you wish to be removed from the RADC mailing list, or if the addressee is no longer employed by your organization, please notify RADC (RBC) Griffiss AFB NY 13441. This will assist us in maintaining a current mailing list.

Do not return this copy. Retain or destroy.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: RADC-TR-78-112	2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER	
4. TITLE (and Subtitle): A MICROPROCESSOR BASED PERFORMANCE MONITOR FOR COMMUNICATION NETWORKS,	5. TYPE OF REPORT & PERIOD COVERED: Phase Report
6. PERFORMING ORG. REPORT NUMBER: N/A	
7. AUTHOR(s): M. S. Callow, R. A. Meyer, C. Crist, A. Black	8. CONTRACT OR GRANT NUMBER(s): F30602-75-C-0082
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Clarkson College, Potsdam NY 13676	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 920W 9561 0015 00
11. CONTROLLING OFFICE NAME AND ADDRESS: Rome Air Development Center (RBC), Griffiss AFB NY 13441	12. REPORT DATE: May 1978
	13. NUMBER OF PAGES: 201
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office): Same	15. SECURITY CLASS. (of this report): UNCLASSIFIED
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE: N/A

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

Same

18. SUPPLEMENTARY NOTES

RADC Project Engineer: Jacob Scherer (RBC)

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

Communications Systems
Systems Monitoring
Microprocessors

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report shows how the microprocessor based monitoring system (M-ATEC) proposed for the FKV area of the EDCS may be extended to include those functions of ATEC not previously included. It also describes ways to reduce the telemetry load by including some functions at present implemented in the PATE control computer. Finally the report describes a microprocessor based version of the eye pattern monitor. Specifically this report describes the addition of a local alarm display and local implementation of statistics calculations

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

and parameter alarm level testing. The report gives details of a complete system, including hardware, software and memory organization, for the task of performance monitoring at the local stations of the FKV link.

PREFACE

This effort was conducted by Clarkson College under the sponsorship of the Rome Air Development Center Post-Doctoral Program for Rome Air Development Center. Mr. Charles Meyer, RADC/DCLD was the task project engineer and provided overall technical direction and guidance.

The RADC Post-Doctoral Program is a cooperative venture between RADC and some sixty-five universities eligible to participate in the program. Syracuse University (Department of Electrical Engineering), Purdue University (School of Electrical Engineering), Georgia Institute of Technology (School of Electrical Engineering), and State University of New York at Buffalo (Department of Electrical Engineering) act as prime contractor schools with other schools participating via sub-contracts with prime schools. The U.S. Air Force Academy (Department of Electrical Engineering), Air Force Institute of Technology (Department of Electrical Engineering), and the Naval Post Graduate School (Department of Electrical Engineering) also participate in the program.

The Post-Doctoral Program provides an opportunity for faculty at participating universities to spend up to one year full time on exploratory development and problem-solving efforts with the post-doctorals splitting their time between the customer location and their educational institutions. The program is totally customer-funded with current projects being undertaken for Rome Air Development Center (RADC), Space and Missile Systems Organization (SAMSO), Aeronautical System Division (ASD), Electronics Systems Division (ESD), Air Force Avionics Laboratory (AFAL), Foreign Technology Division (FTD), Air Force Weapons Laboratory (AFWL), Armament Development and Test Center (ADTC), Air Force Communications Service (AFCS), Aerospace Defense Command (ADC), HQ USAF, Defense Communications Agency (DCA), Navy, Army, Aerospace Medical Division (AMD), and Federal Aviation Administration (FAA).

Further information about the RADC Post-Doctoral Program can be obtained from Mr. Jacob Scherer, RADC/RBC, Griffiss AFB, NY, 13441, telephone Autovon 587-2543, Commercial (315) 330-2543.

ACKNOWLEDGEMENTS

The author wishes to thank his adviser, Dr. Stephen Crist, for his valuable guidance and encouragement without which this work would not have been possible.

He would also like to thank Dr. Bruce Black for his depth of insight into the eye monitor and his help with the simulation system.

Thanks are also due to Allen Heath for his text editor used for preparing drafts and to Vicki Clarke and Donna Boyea who typed the final manuscript.

A final thanks goes to Jack L. Dineley whose help and guidance made it all possible.

CONTENTS

CONTENTS

CONTENTS

LIST OF TABLES

LIST OF FIGURES

1 INTRODUCTION

1.1 General

With the increase of worldwide communications business, great interest is being taken in the reliability of communications hardware. Since no equipment can be made completely free from failure, methods of monitoring which can detect trends leading to failures are of particular interest. Such a system should be able to detect equipment degradation or failure before they can cause communications loss.

Such a system, known as automated technical control, or ATEC, was developed for a communications network in Europe operated by the NATO alliance. This network carries both voice and data communications and is frequency division multiplexed (FDM). A plan of the network is shown in Figure 1. With recent developments in digital communication systems it was decided that a small part of this network should be converted to pulse code modulated (PCM) time division multiplexed (TDM) operation for test purposes.

Since the monitoring requirements of a digital system are different from those of an analogue system, a study was commissioned on the changes which would be required to the ATEC system to meet these needs [1-3]. Because of the advantages of increased capability and flexibility with reduced cost offered by microprocessors a study was also initiated on the use of a microprocessor for the monitoring task. This study resulted in a scheme labelled M-ATEC which is described in RADC report TR-77-218 [4].

This report describes further work on the M-ATEC system. It will show how the capabilities of the M-ATEC equipment may be increased without adding significantly to the hardware costs. The system described in this report will be labelled M-ATEC II to keep in line with previous nomenclature.

1.2 Background

1.2.1 History of the Project

In 1974 the Frankfurt, Koenigstuhl, Vaihingen section of the aforementioned NATO European defense communications network was converted to PCM TDM operation. The control centre for this section is located in Stuttgart. The geographic area of the section is shown in heavy outline in Figure 1.1 and the individual stations are shown in Figure 1.2. It has been undergoing continuous testing since then.

The proposed ATEC equipment consists entirely of special purpose hardware. In 1975 the feasability study for the microprocessor based system was performed by Clarkson College under the Rome Air Development Centre's (RADC's) post doctoral program. The study [4] was delivered in February 1977.

1.2.2 ATEC System Outline

The ATEC system is divided into two parts: monitoring equipment at the individual stations and control and system monitoring equipment at the network control centre. Figure 1.3 shows the equipment installed at each place.

The signals monitored fall into three categories: contact closures, TTL voltages and slowly varying analogue voltages. The TTL level signals may be further subdivided into three categories depending upon their speed of repetition. Slow TTL (STTL) signals are those with a repetition rate of less than one per second. Medium speed TTL (MTTL) signals have a repetition rate greater than one per second and less than one per fifty microseconds. Finally fast TTL (FTTL) signals have a repetition rate greater than one per fifty microseconds.

The site equipment includes an alarm scanner which monitors up to forty on/off alarm switches. This has its own local display board which is a panel having an LED indicator for each alarm monitored plus lamps indicating whether or not any particular alarm is major. It also has an acknowledgement feature which allows personnel at the station to indicate that they have noticed the alarm. Until they have done so the lamps indicating which alarm has been triggered will flash.

There are also a baseband monitor, which monitors the quality of the received signal by means of its eye opening, and an analogue voltage scanner which reads up to thirty-two voltages, including three from the baseband monitor, and converts them to digital values. Finally there is a measurement aquisition controller (MAC) which governs the activities of the analogue voltage scanner and interfaces between it and the control centre computer. Another important function performed by the MAC is an event per unit time counter (EPUT) which is used for measuring the MTTL and FTTL signals.

The equipment required at the control centre includes a remote alarm display for each station, giving the same information as the local display, a master alarm display (MAD) which shows the alarm status of every station, and a computer which is known as the programmable ATEC terminal element (PATE) together with its CRT display and keyboard.

The control centre is linked to the network stations by two telemetry channels. One channel is for the alarm scanner data; it is a simplex channel from the stations to the control centre. The bit rate of this channel is 75 bits per second. The other channel is a half duplex link between the MAC's and the control centre with a bit rate of 150 bits per second.

1.2.3 M-ATEC System Outline

The site equipment required for the M-ATEC system is shown in Figure 1.4. It consists of the baseband monitor and a single microcomputer which carries out the functions of the alarm scanner, the analogue voltage scanner and some of the functions of the MAC.

As the study which resulted in the M-ATEC system was intended only to show the feasability of using microprocessors, no work was done on the transfer of the recorded data from the microcomputer to the control centre. For this reason no interfaces are shown in Figure 1.4.

The microcomputer hardware is shown in Figure 1.5. It consists of the central processor (CPU), two peripheral interface adapters (PIA's), one asynchronous communications interface adapter (ACIA) and read only (ROM) and random access memory (RAM), all interconnected by address and data busses as shown.

The interfaces to the CPU through the PIA's are shown in Figures 1.6 and 1.7.

Figure 1.6 shows the devices served by PIA number one. There is a time-base generator (TBG) which allows the EPUT counting function to be performed by providing an interrupt at preset intervals. Upon receipt of these interrupts a counter is read which gives the number of events of that kind which have occured in that time interval. For the FTTL signals there are hardware counters while for the MTTL signals the count is performed in software on an interrupt basis.

This PIA also acts as the interface for the contact closure and STTl signals which are treated identically. There are five tri-state buffers, each of which can switch eight signals onto a bus which goes to the PIA and hence to the processor data bus. The processor enables each buffer in turn and reads the signals which are present. Two of the buffers have flip-flops at their inputs to latch those signals which, though having a low repetition rate, only appear momentarily. The processor clears these flip-flops after reading the data.

The MTTL signals are counted through this PIA. When a pulse arrives it causes an interrupt which reads which of the two MTTL signals caused the interrupt and then increments an appropriate internal counter. These counters are read when a TBG interrupt occurs.

Figure 1.7 shows the devices served by PIA number two. This again has five tri-state buffers to switch different signals onto a bus on command from the processor. Four of these buffers have eight-bit counters at their inputs for counting the FTTL signals while the fifth has an A/D converter for reading the analogue voltages. Thirty-two voltages can be selected by the processor using the analogue multiplexer. The analogue voltages are read as part of the regular scanning routine while the counters are read on receipt of the TBG interrupts.

As it was not part of this study, the control centre hardware is unchanged.

1.3 Scope of This Work

1.3.1 General

There are two main objectives behind the expansion of the capabilities of

the M-ATEC system which will be described in this thesis. The first is to expand the system to include those functions of ATEC which, though important in a digital communications network, are not fulfilled by M-ATEC. The second is to reduce the quantity of data being transmitted to the PATE, so that it can control more sub-stations, or nodes, than before.

In line with these objectives several new tasks were proposed to RADC as topics for investigation. These were generation of test tones and loopback of duplex lines for testing, local implementation of the statistics calculations, local checking of parameters for correct operating level, application of a microprocessor to the baseband monitor, interfacing the system to the PATE and provision of a local alarm display. In addition, should more than one microprocessor be necessary, the division of tasks between them must be considered.

1.3.2 Statistics Analysis

In the ATEC system the values of all the parameters measured are transmitted back to the PATE where statistical analysis is done. Approximately forty different measurements are made every thirty seconds, all of which must be transmitted. The PATE calculates the mean and the standard deviation of the values of each parameter for every hour. When it has twenty-four hourly values it then calculates the mean and standard deviation for the day. At the end of every month it finds these values for the month.

The microprocessor should easily be able to do the hourly statistics calculations and transmit the values to the PATE. In order to prevent the accumulating totals of the parameters reaching unmanageable proportions, the statistics should be updated continuously rather than waiting until the end of the hour. Also, since the PATE already has a clock, the simplest method of telling the microprocessor that an hour has elapsed would be to transmit an interrupt requesting the statistics.

1.3.3 Parameter Level Checks

In addition to the hard wired alarms of the alarm scanner, the voltages

and other parameters measured by the rest of the system must be checked to see if they are within operating limits. At present this check is done by the PATE, which consequently requires that all these measurements be transmitted to it.

In keeping with the effort to reduce the quantity of data on the telemetry channel and to allow the PATE to handle more nodes, this test should be done locally. In the ATEC system one of five operating levels is signalled for each parameter. These levels are red high, amber high, green, amber low and red low. These tests can easily be accomplished by the microprocessor.

1.3.4. Baseband Monitor

The baseband or eye pattern monitor provides an important test of the quality of the radio links in a digital system. The equipment at present needed for this test is complex and partly digital in nature. The present circuit shown in Figure 1.8 and is described in detail in reference 2. As can be seen in Figure 1.8, the circuit consists of a fast parallel A/D converter with additional circuits controlling the voltage levels on the comparators. The received signal can have one of three levels at the sampling instants. These levels are zero, a positive outer level and a negative outer level. The voltage levels on the comparators are controlled such that a certain percentage of the outer level samples fall either between the 2d and 2d-b or the -2d and -2d+b levels shown. In addition the amplitude of the received signal is controlled by the AGC such that fifty percent of the outer level samples fall outside the 2d and -2d levels.

The present circuit has three outputs which are voltages representing received signal level, average eye opening and number of excessive noise bursts or hits (samples between the d and Nd levels). This circuit examines every sample of the incoming data, but the microprocessor does not have the speed to do this. However, the only purpose for which it is essential to look at every sample is the hit sensor. Therefore, the present hit sensor and counter should be retained.

The microprocessor can sample the flip-flops and from these samples determine whether or not the correct percentage is falling between the 2d and

2d-b levels and also the average eye opening. The only interfacing required for this will be D/A converters. However a microprocessor dedicated to this task would be required and its cost effectiveness must be decided. Since the cost of an eye monitor of this type would be approximately $30,000, a microprocessor should prove cost effective.

1.3.5 Local Alarm Display

The M-ATEC system has no facility for displaying the results of the alarm scan either locally or remotely, both of which are done in ATEC. One possible way to provide these displays is to program the microprocessor to output the alarm state in the message format required for the telemetry channel to the remote displays and use a remote display at the local site as well as at the control centre.

The acknowledgement facility will have to be provided, but this can be done simply by connecting a switch to one of the alarm ports of the processor and then checking this in software.

1.4 Processing Time

In the M-ATEC system almost all the available processing time is used in processing the MTTL signals because of the method of handling them. This is discussed briefly in section 1.2.3.

When an MTTL pulse arrives it is latched by a flip-flop and an interrupt is generated which causes the flip-flop to be read and cleared and an appropriate internal counter to be incremented. When these signals are occuring frequently, almost all the processor's time is used in servicing these interrupts.

In order to free processing time for other functions, the MTTL signals should be counted in an external counter in an identical manner to the FTTL signals. The TBG already provides the necessary timing signal for reading two extra counters as it provides the interrupt signals to read the two internal counters presently used for the MTTL pulses.

2 DATA REDUCTION

2.1 Requirements

The PATE in the ATEC system receives all the samples of every analogue voltage and TTL counter parameter. It uses these values for statistical analysis and alarm level testing. With the intelligence provided at the individual stations by the M-ATEC II system it is advantageous to do these things locally. To do so the microcomputer must do three things. These are alarm level testing, calculation of mean and variance for the analogue voltages and calculation of the mean for the TTL counter values. The resultant data reduction from one value every thirty seconds to two values an hour for each parameter is very large.

This data reduction has two advantages. At present, since every sample is being stored, an enormous amount of data is being collected at the control centre. It is so large that it is leading to data base management problems. A recursive method of calculating the mean and variance based upon the hourly values to be received from M-ATEC II would limit the size of the data base. In addition, the combination of the reduction of data on the telemetry channel plus the reduced computational load on the PATE will allow the same control centre hardware to handle more stations than was previously possible. The reduction in bandwidth on the order wire also allows the same telemetry channel to handle more nodes.

The M-ATEC II statistics routine must therefore provide values of the mean and variance of the samples of the analogue voltages for each hour which the PATE will then use to calculate the daily and monthly values. The transmission of these values could be initiated either by the PATE or by the M-ATEC II microcomputer. Since the PATE has a clock already it may easily be programmed to send an interrupt to the M-ATEC II microcomputer once an hour.

The alarm level testing function of the PATE must also be implemented on the M-ATEC II microcomputer if the data reduction is to be achieved. The PATE,

as mentioned in section 1.3.4, checks every parameter for one of five operating levels. The levels are green, which is the usual operating level, red and amber high, and red and amber low. This test is done simply by storing the alarm levels and making comparisons against these constants.

2.2 M-ATEC II Statistics Routine

2.2.1 Algorithm

The M-ATEC II system is expected to sample every parameter about thirty times a second resulting in the collection of about 110,000 samples of every parameter in an hour. The traditional way of calculating means and variances is to use the expressions in equations 2.1 and 2.2 below.

$$m = \frac{\sum_{i=1}^{n} x_i}{n} \qquad -(2.1)$$

and

$$\sigma^2 = \frac{\sum_{i=1}^{n} (x_i)^2}{n} - \frac{\left(\sum_{i=1}^{n} x_i\right)^2}{n^2} \qquad -(2.2)$$

where n is the number of samples.

x_i are the values of the samples.

These equations require that the sums of the samples and of the samples squared be maintained. If one hours worth of samples were taken, the resulting numbers would have to be handled by floating point arithmetic routines. However, the same recursive system proposed for the PATE would prevent the numbers becoming so large because only a few samples would be collected before a calculation of mean and variance took place.

The operation of such a system is shown in figure 2.1, which shows how the individual samples are handled to give the values for one hour. It works as follows. K samples are collected and the appropriate sums are found. The mean and variance are then calculated by equations 2.1 and 2.2 and the values

are stored while the previous sums are cleared. A new block of K samples is then collected and the mean and variance of them are calculated. When N blocks of K samples have been collected the mean and variance are calculated for all N*K samples and this value is stored. Once an hour, upon receipt of an interrupt from the PATE, the values of mean and variance for the blocks of N*K samples will be combined to give values of mean and variance for the hour. If N is 256 and K is 128 then values of mean and variance for approximately three blocks of N*K samples will be available for calculating the mean and variance for the hour. In order to use this recursive system expressions for the mean and variance of a number of samples, in terms of the previous values of these quantities for smaller blocks of the same samples, must be derived. Suitable expressions are shown in equations 2.3 and 2.4 below.

$$m_{ov} = \frac{\sum_{j=1}^{q} m_j}{q} \qquad - (2.3)$$

and

$$\sigma^2_{ov} = \frac{\sum_{j=1}^{q} \sigma^2}{q} + \frac{\sum_{j=1}^{q} m_j^2}{q} - \frac{(\sum_{j=1}^{n} m_j)^2}{q^2} \qquad - (2.4)$$

where q is the number of equal size blocks for which q values of mean and variance were calculated.

The derivation of the expressions is shown in Appendix A.

The algorithm for the statistics routine based on these expressions is shown in figure 2.2. This gives the sequence of events each time a sample of one of the thirty-two parameters is taken. Normally as each sample is taken its value is added to the sum. Also, the value is squared and this square is added to the sum of the squares.

2.2.2 Implementation

The algorithm just described must be implemented in software for the

microcomputer. Because of the author's previous experience with the device, it was decided to write the programs for the MOS Technology MCS 6502 microprocessor instead of the Motorola M6800 used in the M-ATEC system. The two processors are hardware compatible so that the same interfaces may be used as were used in M-ATEC.

In order to implement the routine described, arithmetic subroutines to add, subtract, multiply and divide are required. The subroutines will work on certain fixed locations which will be called argument registers. The maximum size of the arguments is four bytes and the routines have been written to handle any size up to this. The registers are ADD (addend), AUG (augend), and MULP (multiplier). MULP is two bytes long and the other registers are four bytes long. These are loaded with the arguments before the subroutine is called and the result is returned in AUG. All these registers are located in page zero. Flow charts for the routines are shown in Figures 2.3-2.6 and assembly language listings for the MCS 6502 are shown in Appendices B-E.

When a sample is taken it will be placed in a location TEMPI in page zero and will be passed to the statistics routine. Figure 2.7 shows a flow chart for a program which performs all the necessary calculations and table 1 shows the data which must be stored for each parameter. To simplify the division, it was decided to choose values of N and K which are integral powers of two so that division may be accomplished by a simple shifting operation. It can be seen that the two sections of the program marked by the rows of asterisks are identical in function though working with different data. A subroutine may be written for these sections provided that the data is arranged properly. A suitable addressing mode must be selected to access all the data.

In addition to the routine just described, a program is needed which will be executed on receipt of the interrupt from the PATE. This program has to calculate the mean and variance for the hour for all thirty-two analogue voltages. It must also calculate the means for the various TTL pulse counts. The regular processing of these counters will be described in section 2.5. Because the interface to the PATE will have to be modified for this system and because

this interface is not part of this work, the values destined for the PATE will be stored in the data area for that parameter.

There are two suitable addressing modes, indexed indirect and indirect indexed addressing. In indexed indirect addressing the contents of the X register are added to the second byte of the instruction to give the address of a location in page zero containing the address of a location elsewhere in memory containing the desired data. In indirect indexed addressing the second byte of the instruction gives the address of a location in page zero which is a pointer to a location elsewhere in memory. The contents of the Y register are added to this location to give the address of the required data.

Indexed indirect addressing requires a pointer in page zero for every byte of data which is to be accessed. This causes great difficulties when the required piece of data is more than one byte long. For this reason indirect indexed addressing is the better mode for this application because a single pointer may be used which is changed each time the routine is called and then the Y register can be adjusted to give the required byte of data.

Using the memory organization shown in Figure 2.9 and indirect indexed addressing the assembly language programs shown in Appendices F-H can be written. Appendix F is the main statistics routine, Appendix G is the subroutine which handles the calculations of the mean and Appendix H is the routine to service the PATE interrupt. Each time the routines are called a pointer to the memory area for the current parameter is passed. Also, a value is passed to subroutine MEAN in the Y register to locate which particular data it is to work with as it is called twice for each parameter. The PATE interrupt routine has its own pointer in page zero to avoid having to save the other pointer when the interrupt occurs.

2.3 Alarm Level Test

The alarm level test is merely a series of comparisons against stored constants. These constants are different for each parameter and may conveniently be stored in the same memory area as is used by the statistics routine.

Indirect indexed addressing may also be used. A flow chart for the program is shown in Figure 2.9. By storing the constants in the same block of memory as the statistics data the same pointer may be used as is used by the statistics routine. The routine terminates as soon as the alarm level is found. If no alarm level is found then a green condition is signalled. The assembly language listing is given in Appendix I.

There are four alarm levels which could be signalled. Red is for failure and amber is for imminent failure. Additional information indicating whether the failure is above or below the normal level is also given. Just as in the PATE interrupt routine the data destined for the control centre is stored in the memory block of the appropriate parameter because of the lack of definition of the telemetry channel.

2.4 Overall Analogue Voltage Scan Routine

2.4.1 Interface

The analogue voltage scan interface is shown in Figure 2.10. It is very similar to that of the M-ATEC system. However because the PIA has to handle more devices than before, namely the MTTL counters, the control decoding has to be modified. Also the microprocessor based eye monitor will output a digital value for the eye opening, instead of an analogue voltage, which will require an additional tri-state buffer which is shown in the figure.

The interface consists of two sixteen input analogue multiplexers, a sample and hold, an A/D converter and two tri-state buffers. PIA2A is used to output control signals which select the multiplexer input and enable the required devices to connect that input to the external data bus which goes to PIA1A and hence to the internal data bus.

2.4.2 Program

The analogue voltage scan routine controls the interface and governs the reading of the samples. It then calls the alarm test and statistics subroutines. It also keeps track of the number of samples so that the statistics

routine knows when enough samples have been taken to make calculations. If the statistics routine were to keep this count, an individual count would have to be kept for each parameter.

A memory map for the routine is shown in Figure 2.12. This shows both the zero page registers and the memory block for each parameter. Figure 2.13 shows a flow chart for the program. Each time the program is run it increments the number of samples count which applies to all the parameters. It also sets the pointer to the memory block for the first parameter and then starts sampling the parameters and calling the alarm test and statistics subroutines modifying the pointer after each parameter has been sampled. After all the parameters have been sampled the number of samples count is checked and if it is K then the counter is cleared. The assembly language listing is given in Appendix J.

2.5 MTTL and FTTL Signal Processing

2.5.1 TTL Interface

The interface for the TTL signals is shown in Figure 2.14. It is controlled through PIA2 as is the analogue voltage scan interface. Both medium and fast TTL signals are counted in external counters in the same way that FTTL signals were counted in M-ATEC. The interface consists of six eight-bit counters each of which may be gated onto the data bus to PIA2A through tri-state buffers.

In addition there is a counter in the eye monitor which counts noise bursts. In ATEC and M-ATEC the value of this counter is represented by an analogue voltage level. In M-ATEC II however, the output of this counter will be read directly by the microcomputer. An extra tri-state buffer is needed for this plus a control line to clear the counter.

In order to obtain a value per unit time for the counts the seven counters must be read at predetermined intervals. A time base generator will be used as in M-ATEC. The TBG in M-ATEC provides an interrupt every 440,000 microseconds so that the counters are read every 2.64 seconds. With seven counters to be read the TBG interval will have to be reduced to 380,000 microseconds so that the counters are still read every 2.64 seconds.

2.5.2 TTL Interrupt Routine

On receipt of the interrupt from the TBC this routine reads and clears the next counter in turn and adds its value to a sum for that counter. The alarm level test subroutine is then called to determine if the value of the counter is an alarm level. Then if K values have been read for a particular counter it calculates the mean. When it has N values of mean it calculates the overall mean. The PATE interrupt routine then uses these values to calculate the mean for the hour.

This program is very much like the statistics routine except that only the mean is calculated. It is not considered worthwhile to find the variance, as the counters will have very small readings when functioning normally but will have large readings when things start to go wrong. As with the statistics routine a set of values must be stored for each counter. The memory map for the TTL interrupt routine is shown in Figure 2.15. Indirect indexed addressing is used as in the statistics program to access these values.

A flow chart for the program is given in Figure 2.16. During initialization of the system the pointer is set to the memory block for the first counter and the counter and the TSB numbers are also set. When the routine is called the statistics pointer is saved and the pointer for the first counter is loaded in its place. The counter is read and cleared and the alarm level check subroutine is called. If enough samples have been taken the mean is calculated. Finally the pointer and counter number are set up for the next counter and the statistics routine pointer is restored. An assembly language listing for the TTL interrupt routine is given in Appendix K.

3 ALARM SCAN DISPLAY

3.1 Requirements

The ATEC alarm scanner scans the on/off alarm conditions and displays them both at the site where the scanner is located and at the control centre. These displays consist of a lamp for each alarm, two lamps indicating either major or any alarm and a lamp which lights until the existence of an alarm has been acknowledged by personnel at the site of the alarm. This is done by closing a switch on the alarm display. This switch is only mounted on the displays at the remote sites and not the ones at the control centre.

When an alarm is triggered, the corresponding lamp on the display flashes together with the non-acknowledge lamp and a lamp indicating the type of alarm which has occured. When the acknowledge button is depressed at the site of the alarm the non-acknowledge lamp goes out and the other lamps become steady.

Any display designed for M-ATEC must have these features. In addition the microprocessor should be programmed to output the data in the format required by the alarm displays at the control centre so that this data can be transmitted there.

The data format accepted by these alarm displays is shown in Figure 3.1. The message consists of two types of characters: begin and data characters. Each character is ten bits long and is separated from the previous character by fifteen bit intervals of mark hold. The begin character has start and stop bits and a non-acknowledge (NACK) bit which is set from the time an alarm occurs until the time it is acknowledged. Then there are bits specifying the type of alarm, some filler bits and the character ends with the string 101. The data character consists of start and stop bits, bits specifying the type of alarm and five bits giving the condition of five of the alarms. The message consists of eight data characters, giving the conditions of the forty alarms, preceded by a begin character. This message is being updated and transmitted continuously.

3.2 Display System Design

The alarm scan display hardware used at the control centre in the ATEC system has all the functions required for the local display except the acknowledgement switch. A local display could therefore be provided simply by using control centre display hardware at every remote site. The only additional hardware then required would be a push-button acknowledgement switch. The rest of the task can be fulfilled in software.

The display hardware takes care of flashing the lamps when the NACK bit is set in the begin character so the task of the software is to read the five eight-bit alarm words, as was done in M-ATEC, and then reformat the data into eight five-bit words. It must then check any alarms which are set to see if they are major alarms and using this information it must build the data characters ready for transmission. Transmission of the data must then be initiated and the begin character constructed before each block of data characters is transmitted.

The data rate of 75 bits per second over the telemetry channel will allow three characters to be transmitted every second. The characters will be transmitted through the microcomputer's serial interface device, the ACIA. A block diagram of this device is shown in Figure 3.2. It shows the various registers within the device and the pin assignments of the package. The ACIA will transmit the eight-bit word which is placed in its output register or it will transmit spaces depending on the contents of its control register. The control register contents also specify how many start and stop bits are transmitted before and after the character and which of the various interrupts are activated. The ACIA will give an interrupt when it has transmitted the contents of the output register. The ACIA can also receive serial data but this facility will not be used in the present application.

As the data rate is so slow compared to the speed of the processor an interrupt driven scheme must be used to control the transmission and allow time for other processing. The operation of the system is shown in Figure 3.3. Normally, the ACIA is transmitting blanks. Every 1/3 seconds or 330,000 microseconds an interrupt is received which causes the contents of the control

register to be changed so that the ACIA stops transmitting blanks. The next character to be transmitted is then placed in the output register and the ACIA starts transmitting it. Transmission continues without any intervention from the processor and when it is completed the ACIA generates an interrupt. This causes the processor to load a new control word into the ACIA which starts it transmitting blanks again.

3.3 Interface

To the processor the ACIA looks like two memory locations, one the control register and one the output register. It has two clock inputs one which governs the transmission data rate and the other the received data rate. In this case, as only the transmitter half of the ACIA is being used, a single 75hz clock must be provided at the appropriate clock input. The ACIA is connected in the system as shown in Figure 3.4. It is connected to the processor address and data buses and has external connections to the alarm display, the modem for the alarm telemetry channel and the 75hz clock.

An interrupt is required every 330,000 microseconds and the simplest way to provide this is to have another time base generator like the one which provides the interrupt to read the TTL counters. Since a different interval is required between interrupts another TBG is required. Since both of the PIA's currently in the system are fully utilized a third PIA must be added. Since there will be other uses for this, namely interrupt priority control, this is no disadvantage. The circuit of the TBG and its interface to the computer through PIA3 is shown in Figure 3.5.

The final piece of hardware is the acknowledgement switch. This is simply interfaced to one of the ports for the alarm scan routine. It should be connected to one of the ports with a flip-flop to latch the occurrence of the switch closure. The path to the processor data bus for this switch is shown in Figure 3.6.

3.4 Display Software

The software consists of three programs. These are a main scanning rou-

tine which reads the data at the alarm ports and builds the data characters for transmission, an interrupt routine which fetches the next character for transmission and loads it into the ACIA output register and an interrupt routine which changes the ACIA control register contents so that it transmits blanks.

The flow chart for the main alarm scan routine is given in figure 3.7. The program reads data from the alarm ports and takes the first five available alarm bits for further processing. If any alarm is set it checks to see whether that alarm is major, then constructs the consequent data character which is stored ready for transmission. The program also checks to see if the acknowledgement switch has been closed and stores this information. Since a begin character must be built later, flags indicating major alarm and NACK must be set or cleared as required. An assembly language listing for the routine is given in Appendix L. A subroutine is used to check for major alarms. Figure 3.8 shows the flow chart for this subroutine. It is called from the alarm scan routine and the number of the alarm word which is to be checked is passed in the X register. The routine simply AND's the alarm word passed to it with a stored constant. A non-zero result indicates a major alarm. The assembly language listing for the routine is given in Appendix M.

Figure 3.9 shows the flow chart for the routine which is executed each time a new character must be transmitted. If the character is a data character then the next one in turn is loaded into the ACIA output register and a new control word is loaded into the control register which causes the word to be transmitted. If the next character is a begin character then this must be constructed using the flags mentioned, and then loaded in the ACIA for transmission to the control centre. The assembly language listing of the program is shown in Appendix N.

Figure 3.10 shows the flow chart for the routine which is executed on receipt of the ACIA interrupt. This simply loads a new control word into the ACIA control register which causes it to transmit blanks. The assembly language listing for the program is in Appendix O.

4 EYE PATTERN MONITOR

4.1 Introduction

4.1.1 The Eye Opening

The diagram of Figure 4.1 represents the eye pattern which would be seen on an oscilloscope if a time exposure were taken of the voltage at the input to the decision circuit of the upper level T1 4000 multiplexer under ideal conditions. At the sampling times the voltage would be at one of three distinct levels, hence this is called a three level eye. The decision circuit will sample the eye pattern at the sampling time and decide whether the received signal is an upper, centre or lower level signal. Additive noise causes the received signals to deviate from the ideal levels thus widening the lines in the vertical direction as shown in Figure 4.2. As the noise increases the images corresponding to the upper, middle and lower levels widen until they become so wide that there is no longer a clear separation between levels. When this happens the decision circuit will begin to make mistakes. The spaces between the various levels at the sampling instants are called the eyes.

The size of the eye openings relative to the distances between centres of the adjacent levels gives a good figure of merit for the performance of the RF channels. This measurement, because of its distinctive format, provides a means of predicting performance as well as giving a measure of present time performance. Also, since any eye monitor would be connected between the radio receiver and the T1 4000 multiplexer, it is useful for fault isolation.

4.1.2 Measuring the Eye Opening

The circuit of the present ATEC eye monitor is shown in Figure 1.8. This circuit is designed to measure the amplitude of the signal perturbations relative to the signal level of the received radio baseband. This is the eye opening. It also tests for bursts in signal perturbations and collects the burst count by use of a counter. This data is used to monitor short duration

radio link disturbances. The monitor also provides a measure of the baseband signal amplitude.

To obtain these measurements the output of the radio receiver is first processed by low pass filtering. The output of this filter is in three level partial response format. This signal will have the format described in section 4.1.1 and shown in Figure 4.2. The amplitude of the signal at the output of the filter is controlled by the AGC such that half of the outer level samples will be larger than fixed internal reference voltages labeled 2d and -2d in Figure 4.2. These levels correspond to the ideal upper and lower levels of the three level signal.

The measurement of the deviation of the received signals from these nominal outer levels is accomplished by generating another threshold for each of the outer levels. For the positive outer level this threshold is labeled 2d-b in Figure 4.2. It is controlled such that a predetermined percentage of the outer level samples will occur between this threshold and the decision threshold d. As the monitored signal becomes increasingly noisy, the deviation of the signals from the outer level increases and the eye closes. As it does so the 2d-b level moves toward d and away from 2d. The magnitude of b, therefore, increases with increasing deviation of the samples from 2d. The negative outer level is monitored in a similar manner.

The circuit of Figure 1.8 achieves this in the following way. The received signal, after filtering, is buffered and supplied to eight voltage comparators. The d and -d comparators are used to determine which of the three levels is received. The zero comparator is used for baud timing recovery. The output of this comparator is used to determine if the zero crossing was early or late with respect to the derived clock and this information is used as the error signal for a phase locked loop. This supplies a clock signal which gives the proper time to sample the partial response signal.

The signal level to the comparators is controlled so that half of the outer level samples are greater than +2d or less than -2d as described above.

The automatic gain control circuitry has a long time constant of approximately 2 seconds so as not to respond to short term signal level variations which are measured by the noise burst counter. The output of the gain controller is fed back to a variable gain amplifier at the input to the filter to provide gain control.

The AGC circuit is shown in Figure 4.3. The mode of operation is as follows. The Q outputs of the +2d and -2d comparators are OR'ed together and fed to the input to the increase gain counter and those of the d and -d comparators are connected to the decrease gain counter. This connection is inhibited by the increase gain line as all four flip-flops will be set if the sample should be above 2d or below -2d. When either of the counters reaches a value of eight it triggers a pulse generator which gives a pulse eight bits wide. This pulse is used to control either the charging or discharging of a capacitor. The voltage across this capacitor is directly proportional to the AGC control voltage.

Similar circuits are apparently used to control the 2d-b and -(2d-b) reference voltages. For the positive outer level the reference voltage is controlled so that the predetermined percentage of the samples occurs between 2d-b and d. The voltage b is averaged with the similar signal for the negative level and this output is the measure of the eye opening. The time constants of the control loops are long enough for a significant number of samples to be collected before a change is made in the reference voltage.

An eighth comparator is used to detect major deviations from the positive outer level. This is done by detecting samples which occur in the narrow band between the decision threshold d and a reference level Nd just below d. The value of N is selected so that, under normal conditions, there will be one threshold violation every few minutes. These violations are counted and the eye pattern monitor provides a DC voltage which is a function of this counter value. The other outputs of the eye pattern monitor are the average eye dispersion and the received signal level.

4.1.3 Use of a Microprocessor

Since the data rate of the received signal is approximately 12.5 MHz a microprocessor can not operate fast enough to read every sample. However the purpose of measuring the eye opening is to find the mean of the dispersion of the received signal at every sampling instant. Therefore the only measurement for which it is necessary to read every sample is that of the hit sensor for detecting noise bursts. It should therefore be possible to replace the AGC and reference voltage control circuits with a microprocessor as shown in Figure 4.4. The comparators, flip-flops and filters etc. of the original system are retained along with the baud timing recovery and hit sensing units. The microprocessor will sample the flip-flops as fast as possible and generate values for the AGC and reference voltages. It will also provide a digital value for the dispersion of the eye. The burst counter can be modified so that its value can be read directly just like the counters for the TTL level pulses.

The calculation of the control voltages is basically a counting job for which a microprocessor is ideally suited. The task can be accomplished with six chips: the CPU, an MCS 6530, which has two PIA type output registers and enough RAM and ROM to contain the necessary work space and programs, a 6520 and some D/A converters. It should be emphasized that this processor is a second one which is completely separate from that which is used for the alarm and analogue voltage scanning.

4.2 Hardware Implementation

4.2.1 Interface Devices

Because the software for the eye pattern monitor will be quite short it will be advantageous to use the MOS technology MCS 6530 peripheral interface/memory device for the interfacing. A diagram of the device is given in Figure 4.5 which shows the various registers and devices within the 6530. It has two eight bit bi-directional output ports, each with its own data direction register to set the direction of the data transfer for each bit. It also has a programmable interval timer, a 1024 x 8 ROM and a 64 x 8 static RAM. The

chip has ten address lines A0 to A9 with internal mask programmed decoding to decide the addresses of the ROM, RAM, timer and I/O ports. For the present application the RAM should be in page zero so that the zero page addressing mode may be used and the ROM should be in pages 2 to 5 with the two I/O ports also in page zero between the RAM and ROM. The internal timer will not be used in this application.

Since four I/O ports are needed in this system an MCS6520 will be used for the other two ports. This is just like the Motorola 6820 PIA. Both these devices are fully described in [7].

4.2.2 Hardware Configuration

The configuration of the block labeled MPU in Figure 4.4 is shown in Figure 4.6. This shows all the interfaces between the microprocessor and both the remainder of the eye monitor and the M-ATEC II microcomputer. The input to the processor is a seven bit word made up of flip-flop outputs. The eighth flip-flop, for the zero crossing detector, is not required for calculating the reference and control voltage levels.

Since the flip-flops are clocked much faster than the microprocessor can read them, a latch is needed to hold the value of the outputs long enough for the microprocessor to read it. The latch is enabled by the Q output of a J-K flip-flop which is clocked by the complement of the recovered baud timing clock. When the processor wants to read the flip-flops it outputs a logic 1 on the remaining line of the I/O port, which goes to the J input of the flip-flop with its complement going to the K input. When the flip-flop is clocked the output becomes a one enabling the latch which will then follow the flip-flop outputs until the enable goes low. Because of the complemented clock the output can only go low at a time when the flip-flop outputs are stable. The microprocessor clears the line to the J and K inputs causing the latch enable to go low as described and then it reads the value at the output of the latch.

The AGC and reference voltage values are stored in the two output

registers of the 6520. When these values are to be changed the microprocessor simply increments or decrements these locations. The outputs are connected directly to three D/A converters the analogue outputs of which are connected to the reference comparators and the variable gain amplifier. It is assumed that the positive and negative outer level dispersions will be the same so that the reference voltages will have the same magnitude but different signs. The sign bit of the positive side D/A will be permanently tied to logic 0 while that of the negative side D/A will be at logic 1.

The remaining port of the 6530 will be used to store the eye dispersion measurement value. This value can be read by the analogue voltage scan routine. At the present time this value is simply the magnitude of b just as in the original monitor. A separate port is used so that if in the future more calculations are done within the eye monitor a port will be available to output this value.

4.3 Eye Monitor Software

In programming the eye monitor it was decided to follow closely the methods used in the hardware version of the automatic gain control. Two counters are set up for the AGC and two for the reference voltage. One counter is used to record events requiring an increase in the controlled quantity and the other one is for events requiring a decrease. Just as with the hardware AGC both positive and negative outer levels will be treated the same.

In the case of the AGC the program works as follows. If the input is above +2d or below -2d then the decrease gain counter is incremented and if the input is between d and 2d or -d and -2d then the increase gain counter is incremented. When either of the counters reaches a value of eight the control value in the output register is incremented or decremented as appropriate. By waiting for eight samples before any action is taken some measure of damping is introduced and the system will not respond to short term fluctuations.

For the reference voltage level a similar system is used. This looks

only at those inputs which occur between d and 2d or between -d and -2d. The intention here is to adjust b so that a certain percentage of these samples fall between d and 2d-b or between -d and -(2d-b). The programming here is similar to that for the AGC except that a different number of samples must be collected above 2d-b before the reference level is raised then is collected below it before the level is lowered so that the desired percentage is obtained.

A flow chart for the program which controls both AGC and reference voltages is given in Figure 4.7. If an input shows a zero level sample then the program simply loops back and reads another sample so only a few samples are missed. If the program has processing to do then more samples will be missed but, because of the averaging effect of the measurement, this should not be a problem. The program listing for the eye pattern monitor is given in Appendix P.

4.4 Simulation of the Eye Monitor

Some question has been raised as to the stability of offset threshold eye pattern monitors such as those described above, so it was decided to undertake a computer simulation of the microprocessor based monitor. This involved writing programs to simulate the functions of the eye monitor and to provide suitable signals as its inputs.

The simulation system which has been designed has a program model for the actual eye monitor which is identical in function to the program designed for the microprocessor. It also contains a program for signal generation which provides either a three level partial response signal generated from a random sequence of binary numbers or a signal with constant level samples. Step changes, additive Gaussian noise, ramp changes, and peridoic and random fading may be added to these signals. They are then passed to the model of the eye pattern monitor which applies a voltage gain to the incoming signal and determines the eye opening.

The output of the program is a graph showing reference voltage, AGC

voltage, actual eye opening and measured eye opening against time. This graph is produced on the high speed line printer. The measured eye opening which is plotted is simply the separation of the 2d-b and the -(2d-b) levels. The actual eye opening is the separation of the positive and negative outer level samples. Both of these curves are for illustration only and are intended to make the results easier to interpret. The eye monitor output is actually the value of b. In order to produce graphs of a suitable size for inclusion in this report the system was designed to also produce a list of the values used to plot the graphs on the line printer. These were transmitted to a remote graphics system where the graphs included in this report were produced.

The first test which was tried was a positive step change. The resulting graph is shown in Figure 4.8. Figure 4.8(a) shows the various parameters used for the test such as the input selected, the details of the interference being added to the signal, and parameters within the eye monitor. All four traces are plotted on the graph of Figure 4.8(b). The four traces are labeled as follows: B is the reference voltage; A is the voltage gain of the AGC; E is the measured eye opening and Y is the input signal level. The unit of the time axis is one bit interval of the 12.5 MHz signal or 80ns. For all the following tests a constant outer level signal was selected rather than the three level partial response signal because other tests showed that using a three level signal had no effect on the results. The preset percentage used is twenty-five.

The system starts in the steady state with the AGC at a gain of one and the eye wide open. At a present time a positive step change occurs in the input. The AGC voltage then drops below one to attenuate the excessive input and it continues to fall until the correct sample balance is regained. The reference voltage remains constant until the AGC has adjusted correctly because until it has done so there will be no samples with values below 2d. When the AGC has adjusted the reference voltage in fact remains at the previous value but it is collecting samples again and it can be seen that the idle noise resumes. There is no closure of the eye which is as expected. This is, of course, an artificial case which is used only to check for stability and it would not occur in reality. As can be seen from the graphs

the eye monitor shows no sign of instability.

A negative step change was tried next and the results of this test are shown in Figure 4.9. In this test the reference level started dropping at the step change and the AGC started increasing. The reference voltage shows some overshoot but the AGC appears perfectly damped. Damping can be varied by changing the numbers of samples which must be collected before changes are made. Again in this test the eye monitor shows no sign of instability.

A negative ramp change was tried next and these results are shown in Figure 4.10. They are very similar to those for the negative step change with both AGC and reference voltages adjusting more slowly than with step change and reference voltage showing some overshoot. Once again there is no sign of instability.

Gaussian noise with a variance of 0.25 was then added over the ramp change. The results shown in Figure 4.11 show the input plotted as a series of discrete points. The eye closes at first as the monitor makes an initial adjustment for the noise. Then it levels off until the ramp starts at which point it starts closing again while the AGC increases. Once again the reference level exhibits some overshoot, though it is only very small. After the ramp ends the eye continues to close because, now that the input signal level is reduced the effect of the noise is much stronger and it is swamping the signal much more. However the monitor remains stable throughout this test.

Finally sinusoidal fading was tried to see if this would make the system become unstable. These results are shown in Figure 4.12. In this case the AGC almost exactly follows the sinusoid while the reference voltage shows only a slight tendency to follow it. The peak value or the reference voltage occurs somewhat after the peak of the sinusoid. It is very small and basically the reference voltage yields a measured value of eye opening which is approximately the mean of the sinusoid of the actual eye opening and the monitor shows no instability.

Other tests were made using three level inputs but the results were no

different. The only effect was that the execution time of the simulator was increased because more samples had to be generated. Various combinations or changes were tried but no condition was found which would cause instability in the eye monitor. It has, therefore, been demonstrated that the offset threshold monitoring concept as used here gives a stable eye monitor with reasonable results.

4.5 Further Development of the Eye Monitor

Further work remains to be done in several areas, such as selecting the best values for the number of samples taken before changes are made to obtain the best damping. There has been very little previous work done on this subject. No references were found which were of any assistance except for [2]. Further work also remains to be done on possible uses of the measured data. Calculations which may be done include finding the bit error rate.

In the ATEC eye monitor the signal level is monitored to detect degradation in the received signal level (RSL) that may not appear as eye dispersion. When the monitor is installed an RSL versus eye dispersion calibration table is generated and stored in the PATE. The value of RSL corresponding to the eye dispersion is compared with the RSL value corresponding to the PCM threshold (-71dBm). The difference is known as the baseband eye margin.

The measured values of eye dispersion and RSL are combined to provide the actual signal to noise ratio of the channel. Knowing the S/N ratio and assuming the additive noise to be Gaussian, the bit error rate (BER) of the channel may be calculated.

All the calculations just described can be accomplished within a microprocessor based eye monitor instead of requiring the PATE. A self contained monitor could be built possibly requiring a second processor which would produce values for channel signal to noise ratio, bit error rate and eye margin. These measurements would be a very good indication of the state of the channel.

5 SYSTEM CONFIGURATION

5.1 System Software

5.1.1 Programs

The software for the system consists of two main programs, four interrupt routines and some subroutines. Table 2 lists all these programs. The routine which initializes the system ends in a loop which then runs continuously calling the two main programs: the alarm scan routine and the analogue voltage scan routine. This program is described in section 5.3.

The interrupt routine TTLINT services the TTL counters. PATINT services the PATE interrupt calculating and transmitting the hourly values of mean and variance. ACAINT services the interrupt from the ACIA, which requests more data, resetting the control word in the ACIA so that it transmits blanks. CHRINT services the interrupt from TBG2, loading the next character to be transmitted into the ACIA output register and setting the control word so that the character is transmitted. The subroutine CCRTN services the contact closure and STTL alarms reformatting the eight bit alarm words into five bit alarm words. The subroutine ANVSCN services the 32 analogue voltages doing alarm level testing and calculating statistics.

The software requires 1536 bytes of RAM and 2411 bytes of ROM. Figure 5.1 shows a memory map for the entire system showing the various storage locations and program locations. The map also shows the interrupt vectors, the stack pointer and the interface locations.

5.1.2 Interrupt Linking

The four interrupt programs must be linked in such a way that the right program is called for each interrupt within the required time limit. Table 3 lists the interrupt routines and the time by which each must be serviced.

The MCS 6502 processor has two interrupt inputs: a non-maskable interrupt (NMI) and an input known as IRQ. The IRQ input may be disabled by a program which must not be interrupted. Each input has a vector which points to the program which is to be executed on receipt of an interrupt at that input. The NMI input has priority over the IRQ input.

TTLINT must not be interrupted by any other routine and it must be serviced immediately. It should therefore be connected to the NMI input which will automatically disable the IRQ input. This will mean that the TTLINT routine can not be masked out by any other routine and that it will have priority over all the other interrupts.

The ACAINT and CHRINT routines must be serviced within 1,333 microseconds so that the changes to the ACIA registers are made by the time the next bit is to be transmitted. These routines may be interrupted by TTLINT and still be completed in time but they may not be interrupted by PATINT as it executes for more than 1.3 milliseconds.

The PATINT routine because of its length must only be allowed to interrupt the main program. But it must not even be allowed to do this during the time that the statistics values are being updated by the statistics routine in the main program. PATINT may be interrupted by any of the other routines.

The necessary interrupt priorities can be accomplished simply by hardware using the unused half of PIA3 as the interface. Figure 5.2 shows the hardware required to implement the scheme. Interrupt 1 (from TBG1) is connected to the NMI input and the NMI vector points to the start of the TTLINT routine. Interrupts 2, 3 and 4 are latched by flip-flops whose outputs are connected to a priority encoder. In addition the flip-flop which latches the PATE interrupt is connected to the priority encoder through an AND gate so that this interrupt may be disabled by the main program when necessary. The outputs of the priority encoder are NOR'ed to the IRQ input and they are also connected to PIA3A. The data read from this port is used to determine which interrupt has occured. The IRQ vector points to a special program which determines which interrupt is to be serviced.

A flow chart for this program is given in Figure 5.3 and the assembly listing is given in Appendix Q. This program reads the output of the priority encoder and uses this value to determine the target location of a jump to the desired interrupt routine. This program also clears the flip-flop ready for the next interrupt. If all the interrupts should occur together then the NMI will be serviced first followed by the interrupt with the highest input to the priority encoder and so on. The individual interrupt routines will disable the IRQ input as necessary.

5.2 System Hardware

Figure 5.4 shows the equipment required at each station of the network for the M-ATEC II monitoring system. There is the microcomputer which accepts as inputs the 40 contact closure/slow TTL signals, the 32 analogue voltages and the outputs of the eye monitor. Other equipment includes the modems for the telemetry channels, the eye monitor and the local alarm display.

The hardware of the microcomputer system is shown in Figure 5.5. It consists of the CPU, three PIA's and an ACIA interconnected by data and address busses. The devices served by the PIA's are shown in Figures 5.6-6.8. Figure 5.6 shows the devices served by PIA1. These are the tri-state buffers and flip-flops for the contact closure alarms and TBG1 which is connected to this port so that it may be loaded initially with the right time interval.

Figure 5.7 shows the devices served by PIA2. These are the analogue voltage scan interface and the TTL pulse counters. This PIA also serves the eye pattern monitor controlling the reading of the value of the eye opening and the noise burst counter.

Figure 5.8 shows the devices served by PIA3. These are TBG2 which is initially loaded through this port and the priority encoder for the interrupts which was described in section 5.1.2.

5.3 System Initialization

When the system is first started up there are several initial tasks which must be carried out before commencing monitoring operations. These include setting up the NMI and IRQ vectors, loading the time base generators with the correct time intervals and setting the various PIA control registers so that the pins are set as inputs or outputs as desired.

Figure 5.9 shows a flow chart of the startup routine necessary to get the system running. The interrupts are first masked and then the ACIA is initialized and programmed to transmit blanks. The interrupt vectors are then set and the time base generators loaded and enabled. The program to load the TBG's are then reset as inputs. Some sort of synchronization with the PATE must be carried out so that it knows when to send the hourly interrupts. Finally all the storage locations must be cleared and the alarm levels and other constants must be stored. The contact closure routine is then called to

generate the first set of alarm words. The interrupt is then enabled and the program enters the main loop calling the main scanning routines. An assembly language program for this routine has not been written as this will be dependent upon the PATE interface. It may for instance be best to query the PATE operator for the alarm levels so that they may be easily updated. There are other items which may require communications with the PATE as well such as specifying which of the contact closure alarms are major.

6 CONCLUSIONS

At the outset of this work several tasks were proposed for investigation. These were generation of test tones and loopback of duplex lines for testing, local implementation of the statistics calculations, local testing of parameters for alarm levels, application of a microprocessor to the eye pattern monitor, interfacing the system to the PATE and provision of a local alarm display.

When two of these tasks were initially considered, a decison was made not to proceed any further with them. The first of these was the loopback of duplex channels. In a digital system the only information which could be gained from such a test would be the bit error rate of the channel. Finding this requires a large amount of data to be statistically valid which, in turn, would require a long time. The same information can be derived from the eye pattern monitor and also from the number of violations of the three level partial response format which are measured in the T1 4000 multiplexer. These tests are done on active channels whereas for loopback the channel must be disconnected. It is, therefore, not worthwhile implementing such a test.

The second task which has not been investigated is that of the PATE interface simply because it is beyond the scope of this work. This interface is involved mainly during the initialization of the system. During this time it would be useful if the PATE operator could specify such things as the alarm levels and which alarms are to be considered major. It would be worthwhile to write a monitor program so that these items can be set by the PATE operator after which the system can be left on its own. Another interrupt could be included which would make it possible for the operator to stop the M-ATEC II system, alter these values and restart it remotely. All that would be required would be a jump to the monitor program when this interrupt occurs. It would also be possible to query the system for particular values if desired. The only other communication necessary with the PATE is the hourly transfer of the statistics values which is also initiated by an interrupt. A data format

must be specified before this transfer can be finalized.

This report has described a system in which all the other tasks mentioned have been implemented. This system replaces the digital ATEC with a reliable digital system offering reduced use of the orderwire bandwidth and the PATE, which means that the same PATE and orderwire can be used to monitor more individual stations than was possible before. This system can also provide remote selection of major alarms, unlike ATEC in which these were hard wire selected. The inclusion of a microprocessor based eye monitor means that direct readings of the signal to noise ratio and bit error rate of the channel could be produced which would give an excellent indication of the condition of the channel.

It is desirable that a test system should provide some indication of its correct operation. A self test routine can be included in M-ATEC II with the provision of a special test program and some additional hardware. For instance extra outputs would be needed to allow the microprocessor to clock the TTL counters to test their correct operation. Further interfacing would also be necessary to allow other parts of the system to be exercised by the microprocessor. Such a test routine could be included as part of the initialization program. A separate program could also be included to provide the special test signals for signature analysis so that, in the event of a breakdown in M-ATEC II, the fault could be traced rapidly using this relatively new technique and the system repaired without the need for board exchange.

The main areas in which further work would be valuable lie in the eye monitor, as described in section 4.5, and in the area of self testing to ensure reliable monitoring and to build operator confidence in the system.

REFERENCES

1. ATEC Digital Adaptation Study - Volume I. Honeywell Inc., St. Petersburg, Florida. October 1976. RADC Technical Report TR-76-302.

2. ATEC Digital Adaptation Study - Volume II. Honeywell Inc., St. Petersburg, Florida. October 1976. RADC Technical Report TR-76-302.

3. ATEC Digital Adaptation Study - Volume III. Honeywell Inc., St. Petersburg, Florida. October 1976. RADC Technical Report TR-76-302.

4. Microprocessor Utilization in Communications System Monitoring. Clarkson College, Potsdam, New York. March 1977. RADC Technical Report TR-77-218.

5. MC6800 Microprocessor Applications Manual. Motorola Semiconductor Products Inc., Phoenix, Arizona. 1975.

6. MCS6500 Microcomputer Family Programming Manual. MOS Technology Inc., Norristown, Pennsylvania. January 1976.

7. MCS6500 Microcomputer Family Hardware Manual. MOS Technology Inc., Norristown, Pennsylvania. January 1976.

NOMENCLATURE

ACAINT.....Data Request Service Program(ACIA interrupt)

ACIA.....Asynchronous Communications Interface Adapter

A/D.....Analogue to Digital

ADDITN.....Addition Program

AGC.....Automatic Gain Control

ALMTST.....Alarm Level Test Program

ANVSCN.....Analogue Voltage Scan Program

ATEC.....Automated Technical Control

CCRTN.....Contact Closure Routine(Alarm Scan Program)

CHRINT.....TBG2 Interrupt Service Program(Character Interrupt)

CPU.....Central Processing Unit

CRT.....Cathode Ray Tube

D/A.....Digital to Analogue

DIVIDE.....Divide Program

EPUT.....Event per Unit Time Counter

EYEMON.....Eye Monitor Program

FDM.....Frequency Division Multiplexed

F/F.....Flip-Flop

FKV.....Frankfurt Koenigstuhl Vaihingen

FTTL.....Fast TTL Level Signals

IRQ.....Interrupt Request

LED.....Light Emitting Diode

MAC.....Measurement Acquisition Controller

MAD.....Master Alarm Display

M-ATEC.....Microprocessor Based Automated Technical Control

M-ATEC II.....Microprocessor Based ATEC, Version 2

MEAN.....Subroutine to Calculate Means

MJACHK.....Major Alarm Check Subroutine

MLTPLY.....Multiplication Program

MTTL.....Medium Speed TTL Level Signals

NACK.....Non-Acknowledge

NMI.....Non Maskable Interrupt

PATE.....Programmable ATEC Terminal Element

PATINT.....PATE interrupt service program

PCM.....Pulse Code Modulated

PE.....Priority Encoder

PIA.....Peripheral Interface Adapter

RADC.....Rome Air Development Center

RAM.....Random Access Memory

ROM.....Read Only Memory

STTL.....Slow TTL Level Signals

STATIS.....Statistics Program

SUBTRT.....Subtraction Program

TBG.....Time Base Generator

TDM.....Time Division Multiplexed

TSB.....Tri-State Buffer

TTL.....Transistor Transistor Logic

TTLINT.....TBG1 Interrupt Service Program(TTL Counter Service)

TABLE 1

QUANTITIES REQUIRED FOR STATISTICS CALCULATIONS

Quantity to be calculated	Quantities required for calculation
1) Mean of 1 block of samples (k samples)	1) Sum of sample values 2) Number of samples
2) Variance of 1 block of samples	1) Sum of samples 2) Sum of $(\text{samples})^2$ 3) Number of samples
3) Mean of a number of blocks of samples (a group of samples) (N*K samples)	1) Sum of means of each block 2) Number of blocks
4) Variance of a group of samples	1) Sum of variances of each block 2) Sum of means of each block 3) Sum of $(\text{means})^2$ of each block 4) Number of blocks
5) Mean of a number of groups of samples	1) Sum of means of each group 2) Number of groups
6) Variance of a number of groups of samples	1) Sum of variances of each group 2) Sum of means of each group 3) Sum of $(\text{means})^2$ of each group 4) Number of groups

TABLE 2

M-ATEC II System Programs

Mnemonic	Use	Size (bytes)	Execution time (μs)
ADDITN	Addition	16	31 + 21 for each additional byte of argument.
SUBTRT	Subtraction	16	31 + 21
MLTPLY	Multiplication	65	364 → 1188
DIVIDE	Division	131	1000
STATIS	Statistics Calculations	419	39,600 → 97,000
MEAN	Mean part of statistics calculations	187	531 → 567
PATINT	Services hourly PATE interrupt	449	18,800
ALMTST	Tests for alarm levels	68	37 → 73
ANVSCN	Analogue voltage scan routine	96	36,600 → 133,600
TTLINT	Services interrupt from TBG1 to read counters	250	304 → 505
CCRTN	Contact closure/STTL Scan routine	220	267 → 496
MJACHK	Checks for major alarms in CCRTN	29	25 → 28
CHRINT	Services TBG2 interrupt, loading next character ACIA	55	63 → 80
ACAINT	Services ACIA request for more data	7	19

TABLE 3

M-ATEC II Interrupt Routines

Interrupt	Service Time	Routine to be Executed
from TBG1	Immediate	TTLINT
from TBG2	Within 1,333 μs	CHRINT
from ACIA	Within 1,333 μs	ACAINT
from PATE	Within 40 ms	PATINT

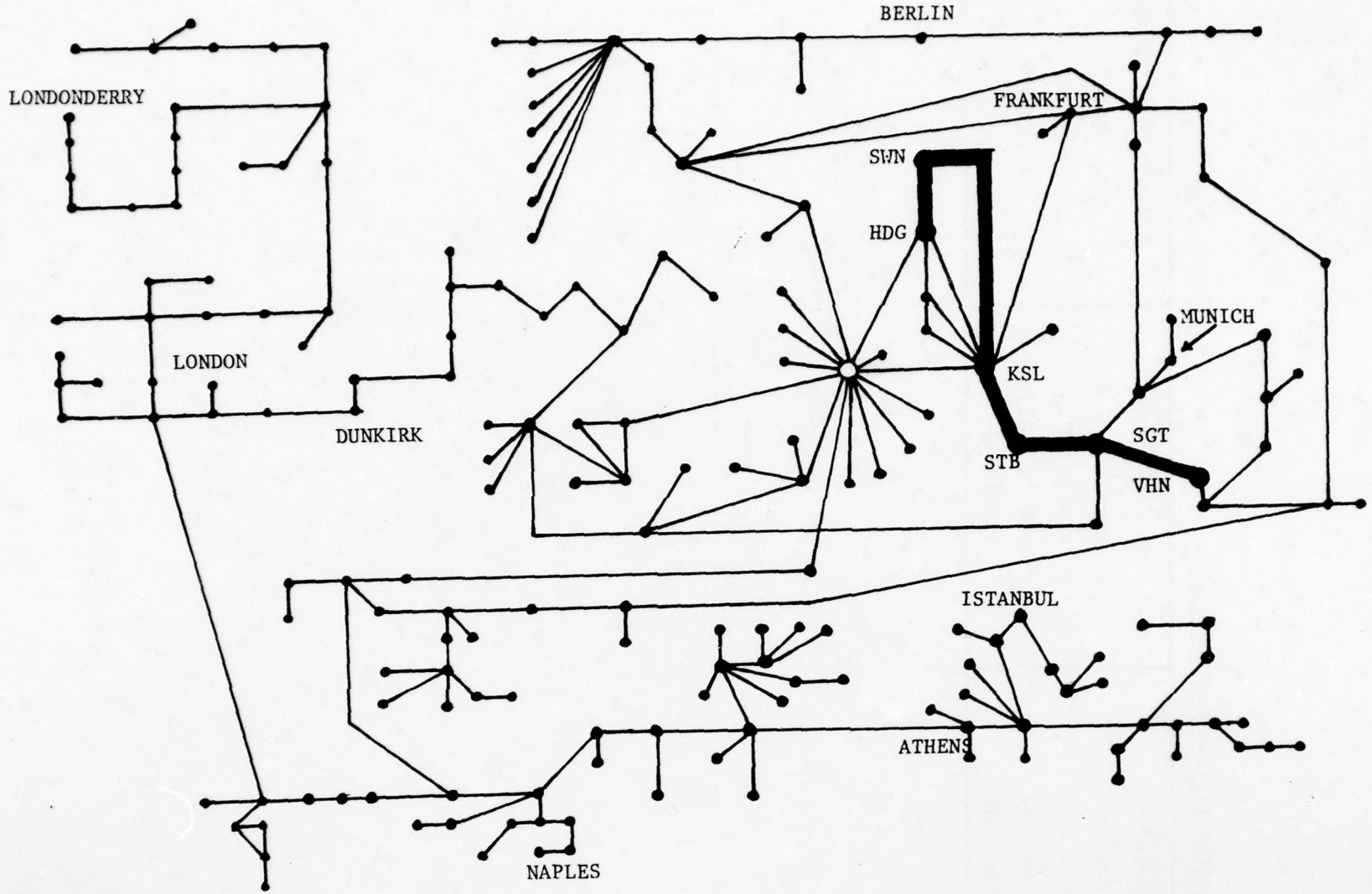


FIGURE 1.1: EUROPEAN DEFENCE COMMUNICATIONS NETWORK

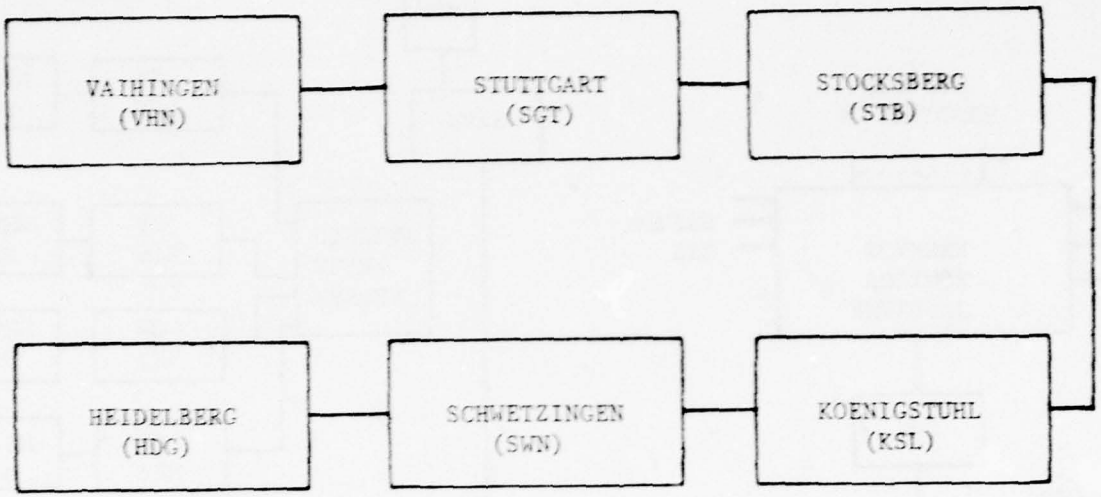


FIGURE 1.2: THE FRANKFURT-KOENIGSTUHL-VAIHINGEN AREA

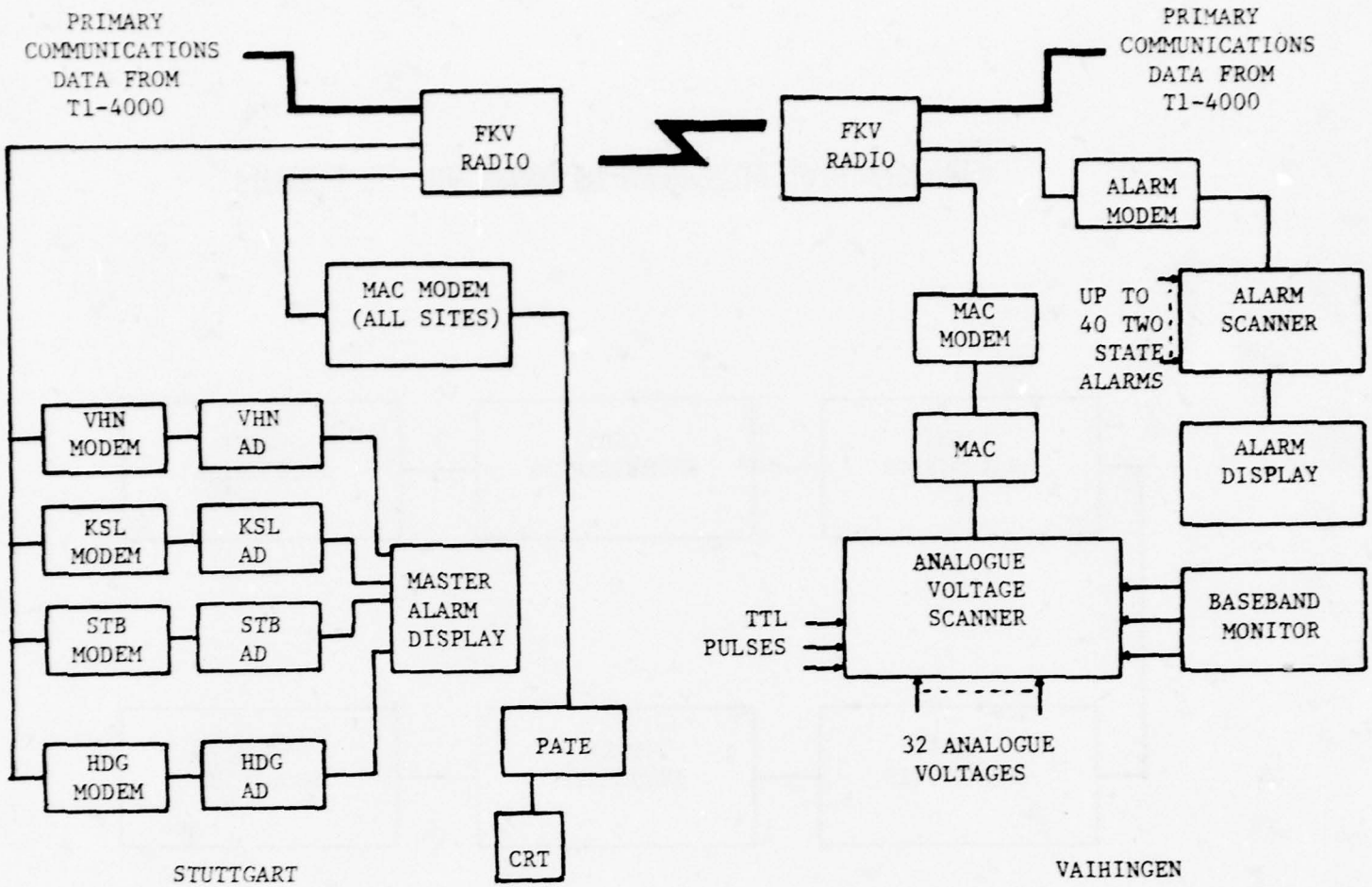


FIGURE 1.3: ATEC CONFIGURATION

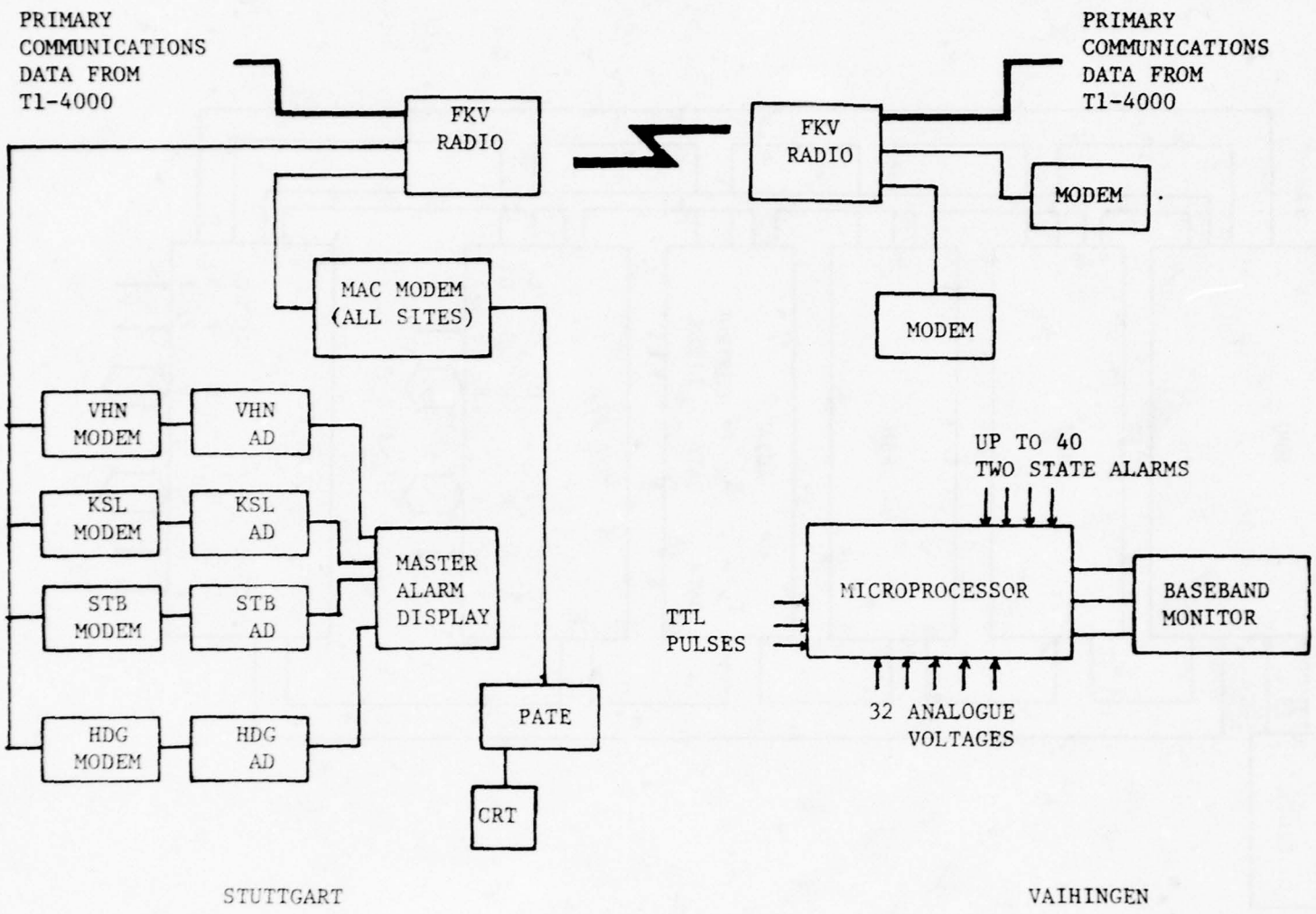


FIGURE 1.4: M-ATEC CONFIGURATION

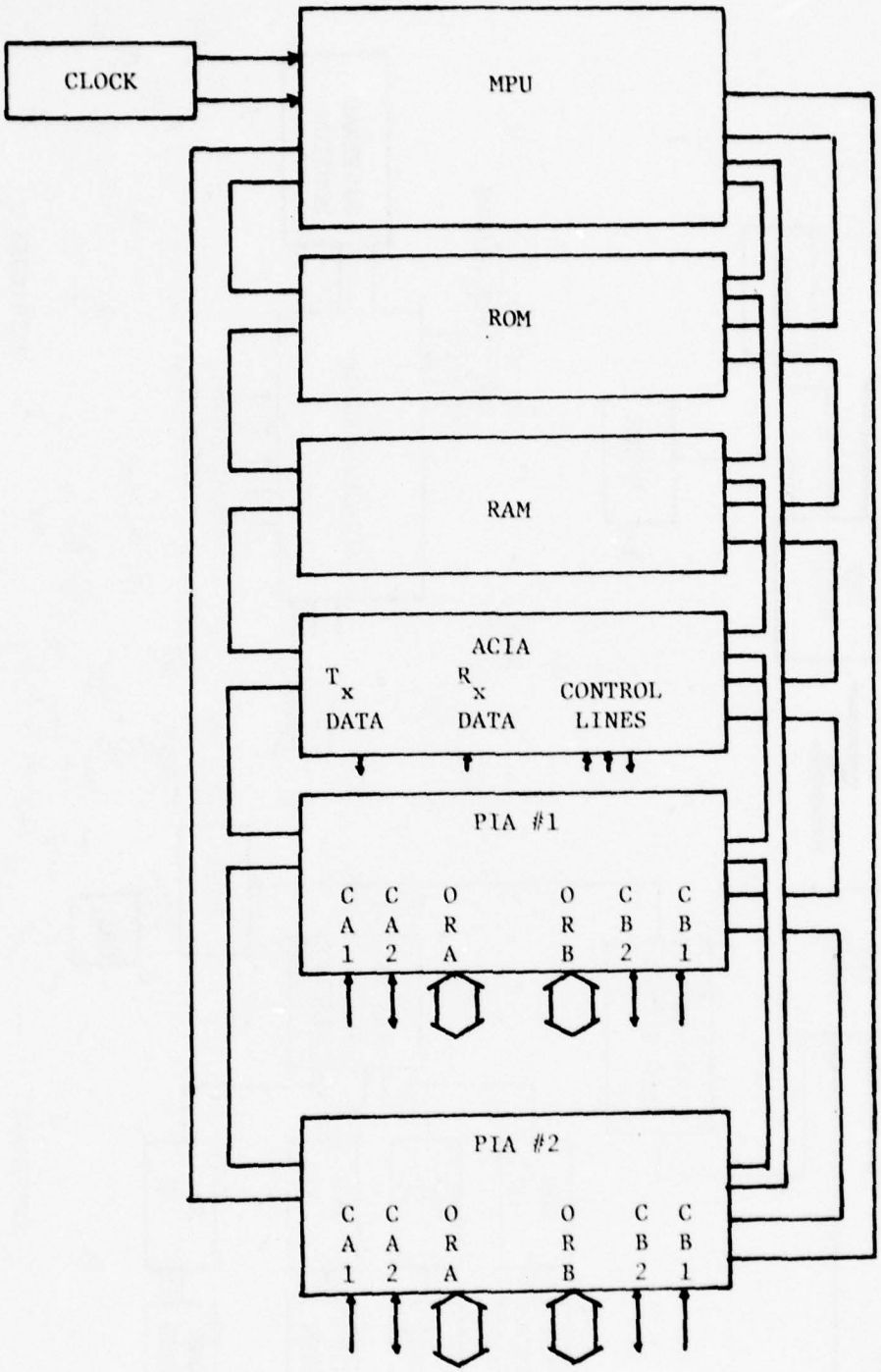


FIGURE 1.5: M-ATEC MICROCOMPUTER HARDWARE

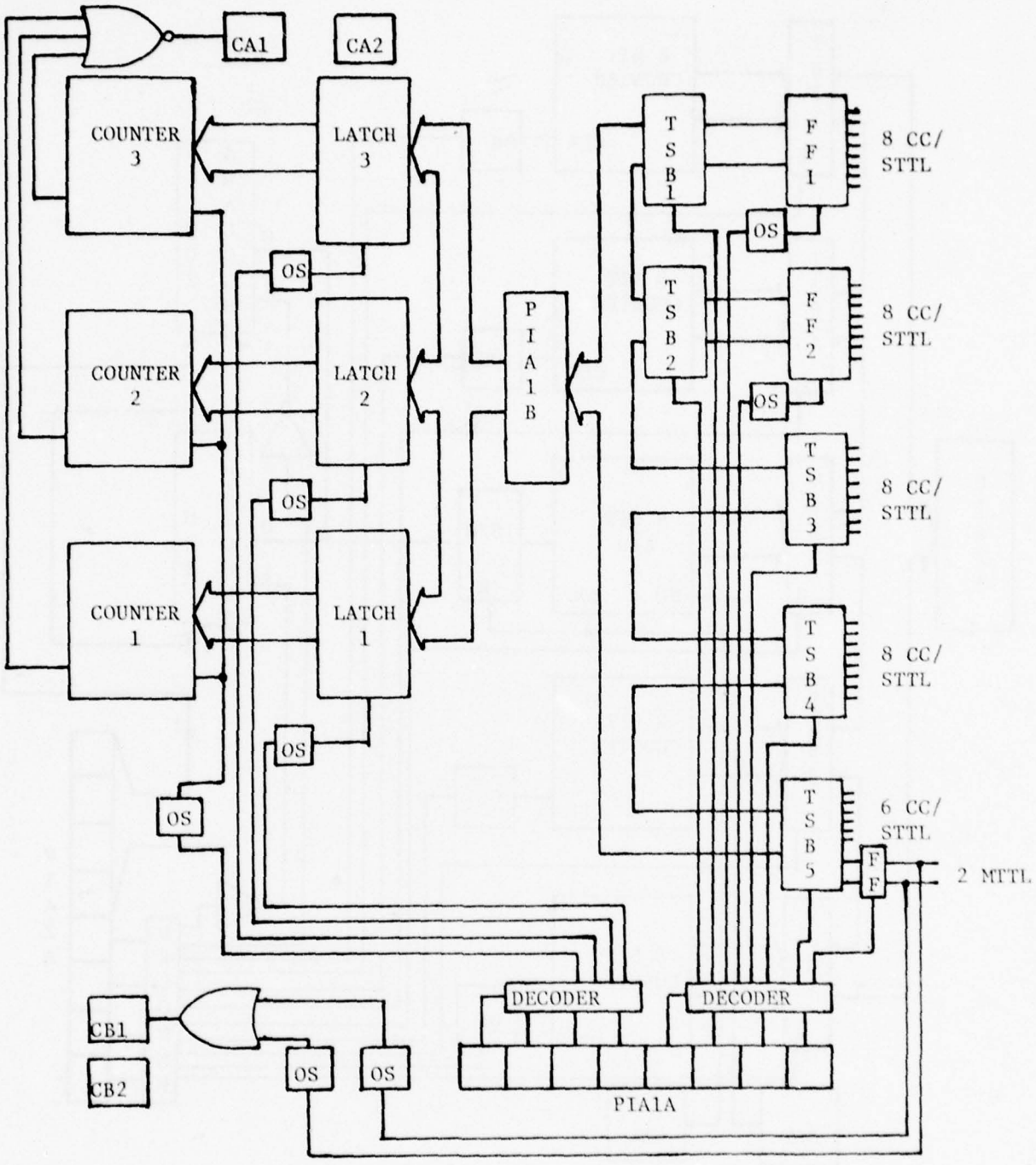


FIGURE 1.6: M-ATEC PIA1 INTERFACE

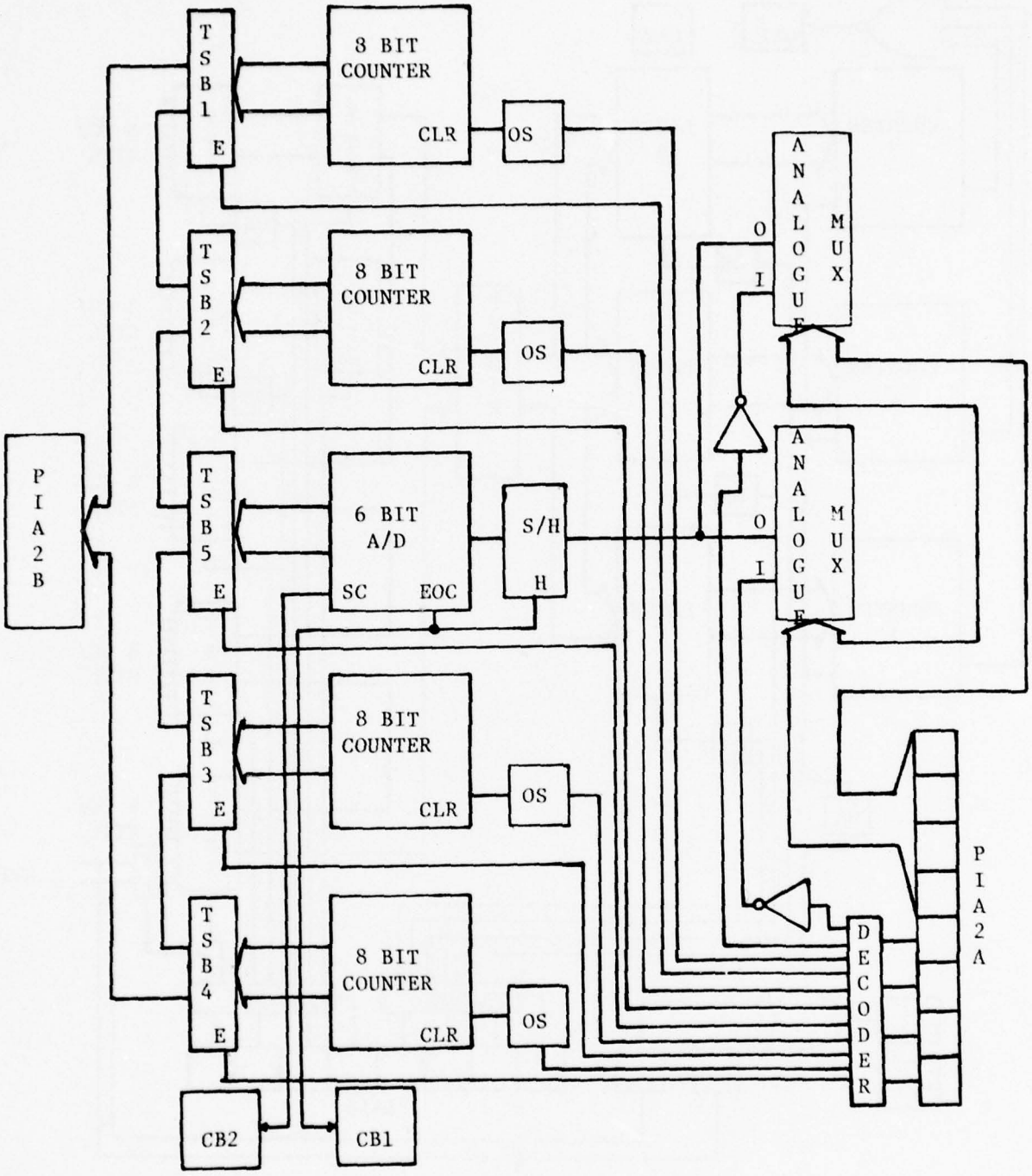


FIGURE 1.7: M-ATEC PIA2 INTERFACE

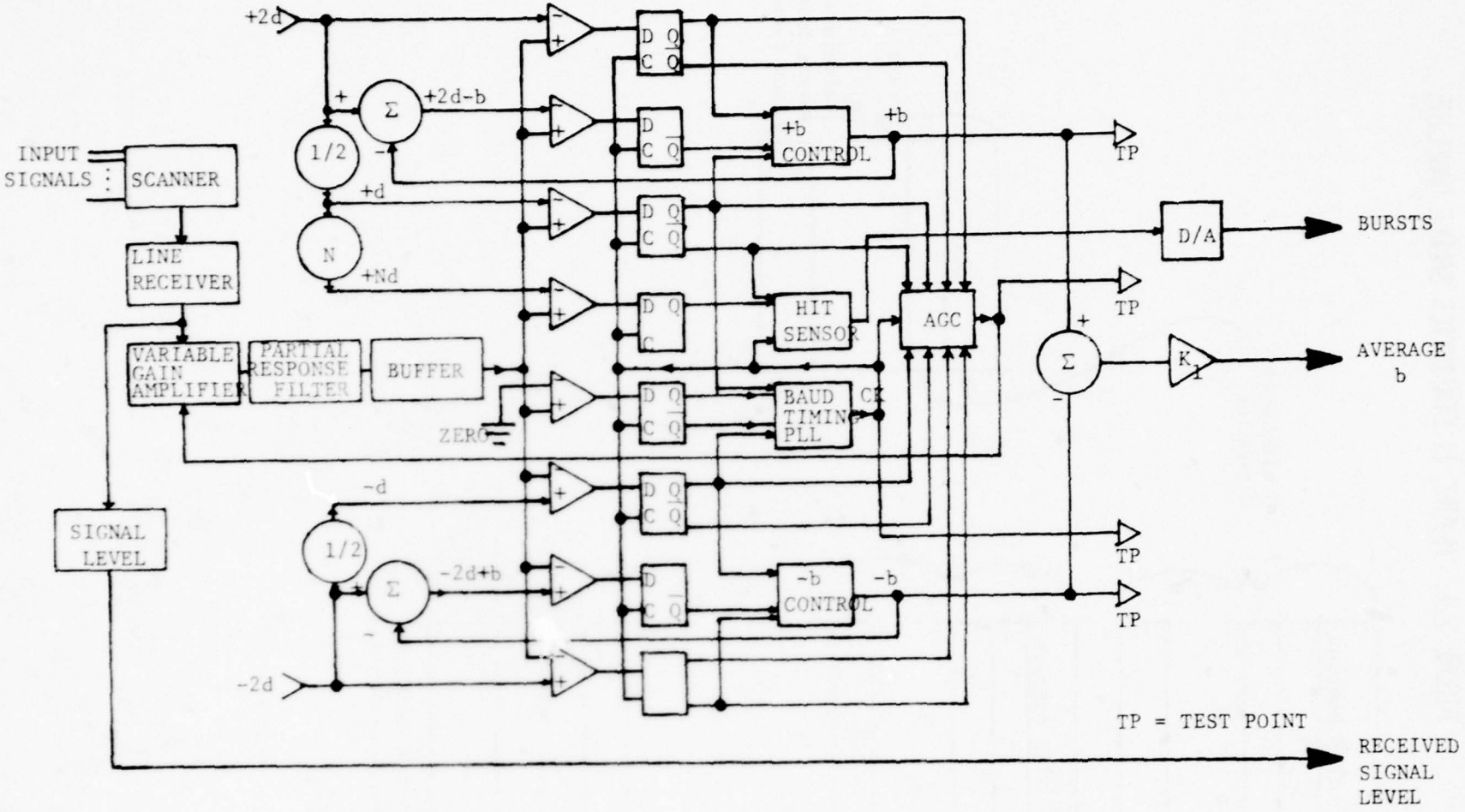


FIGURE 1.8: DIGITAL BASEBAND MONITOR (EYE PATTERN)

FIGURE 2.1: M-ATEC II STATISTICS SAMPLE HANDLING

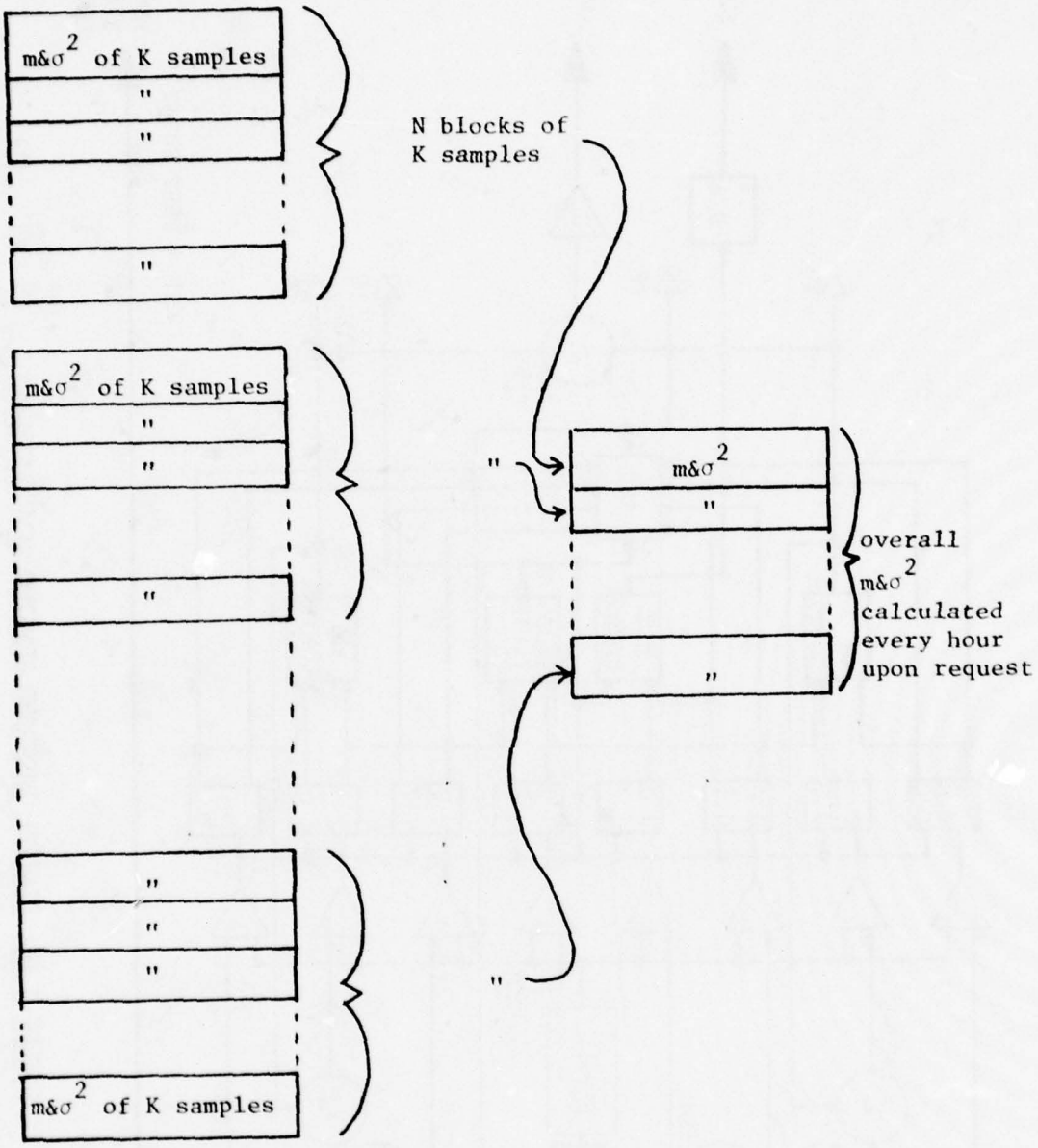


FIGURE 2.2: M-ATEC II STATISTICS ROUTINE ALGORITHM

```
BEGIN
   RECEIVE A SAMPLE
   SUM THIS VALUE
   SQUARE THE VALUE AND SUM THE SQUARES
   IF K SAMPLES OF THIS PARAMETER HAVE BEEN MEASURED
   THEN DO BEGIN
      CALCULATE THE VARIANCE OF THE K SAMPLES
      SUM THIS VARIANCE
      CALCULATE THE MEAN OF THE K SAMPLES
      SUM THE MEAN
      SQUARE THE MEAN AND SUM THE SQUARE
      CLEAR THE STORAGE LOCATIONS FOR THE K SAMPLES
      END IF
   IF N BLOCKS OF K SAMPLES HAVE BEEN COLLECTED
   THEN DO BEGIN
      CALCULATE THE VARIANCE FOR N BLOCKS OF K SAMPLES
      SUM THE VARIANCE
      CALCULATE THE MEAN FOR N BLOCKS OF K SAMPLES
      SUM THE MEAN
      SQUARE THE MEAN AND SUM THE SQUARE
      CLEAR THE STORAGE LOCATIONS FOR THE N BLOCKS
      END IF
   IF AN INTERRUPT IS RECEIVED FROM THE PATE
   THEN DO BEGIN
      CALCULATE THE VARIANCE UP TO THE TIME OF THE INTERRUPT
      CALCULATE THE MEAN UP TO THE TIME OF THE INTERRUPT
      TRANSMIT THESE VALUES TO THE PATE
      CLEAR STORAGE LOCATIONS FOR THE N BLOCKS OF K SAMPLES
      END IF
END
```

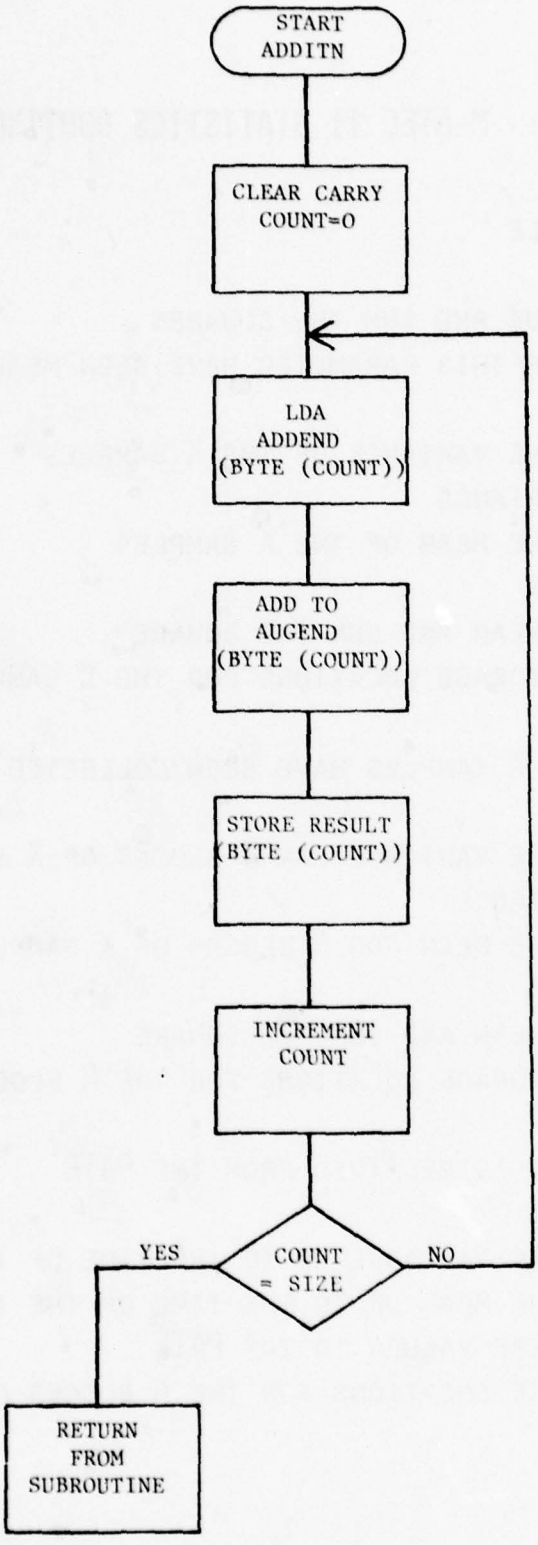


FIGURE 2.3: M-ATEC II ADDITION SUBROUTINE

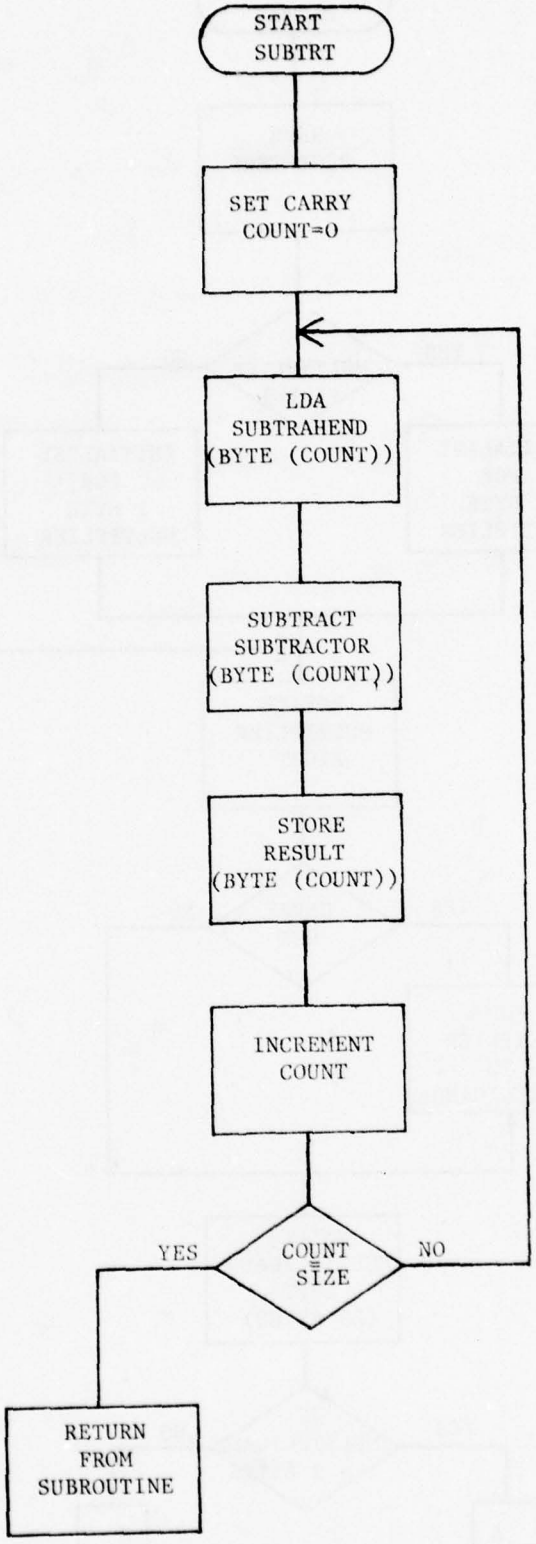


FIGURE 2.4: M-ATEC II SUBTRACT SUBROUTINE

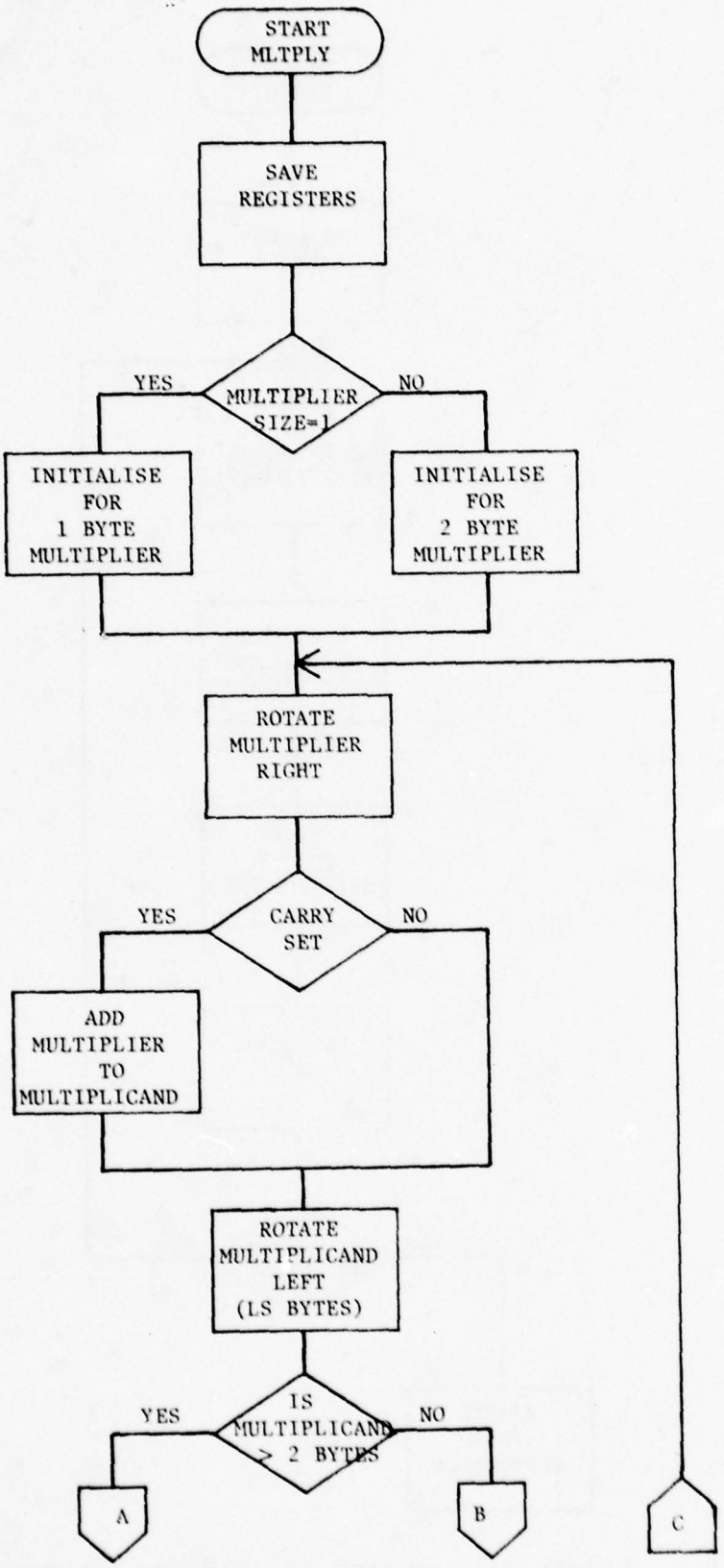


FIGURE 2.5: M-ATEC II MULTIPLY ROUTINE

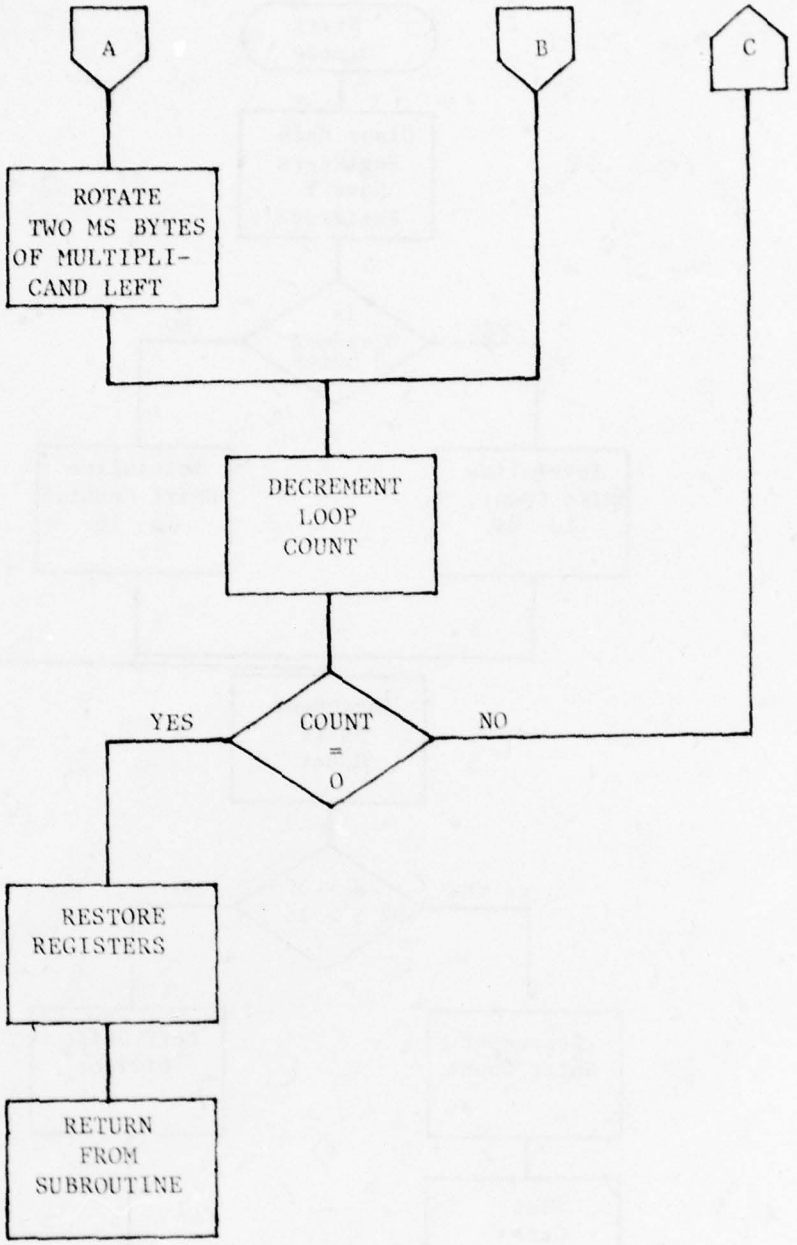


FIGURE 2.5: M-ATEC II MULTIPLY ROUTINE (CONT)

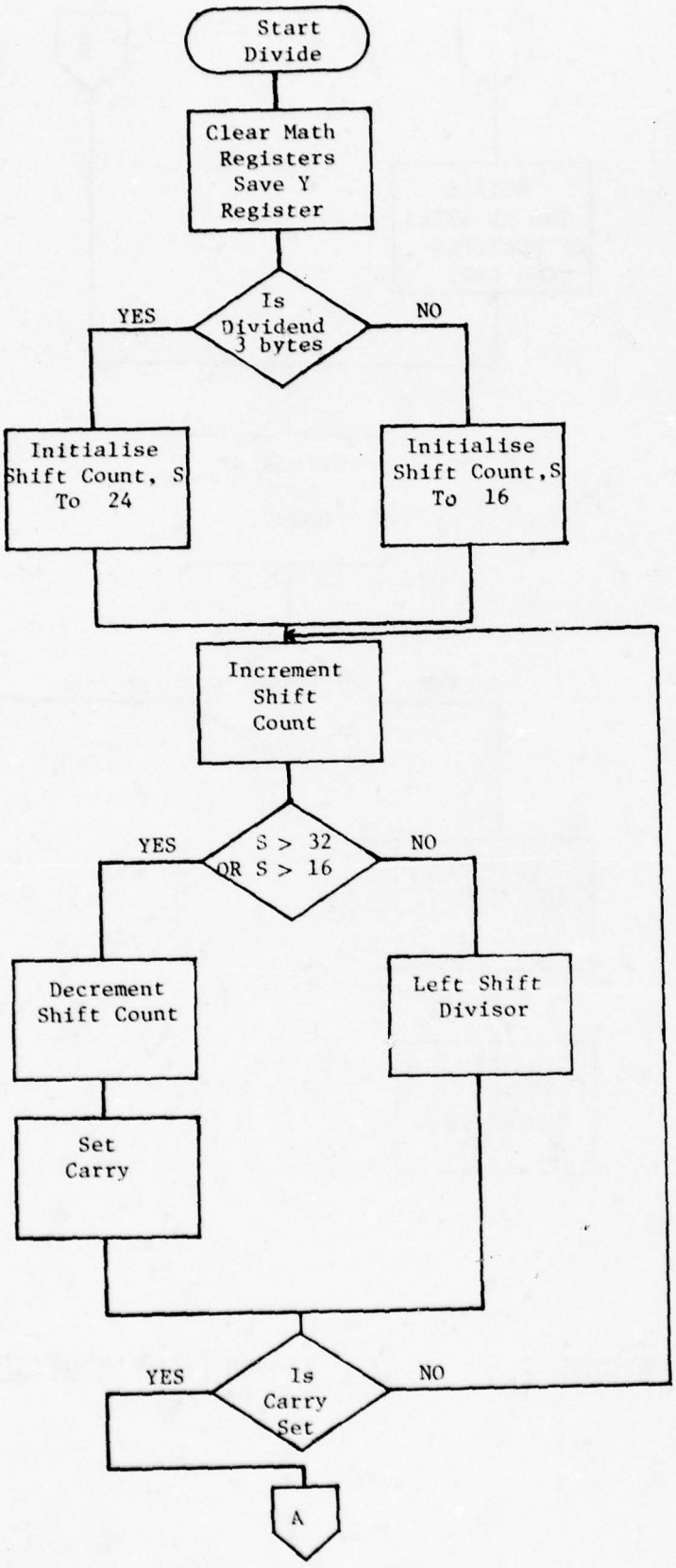


FIGURE 2.6: M-ATEC II DIVIDE SUBROUTINE

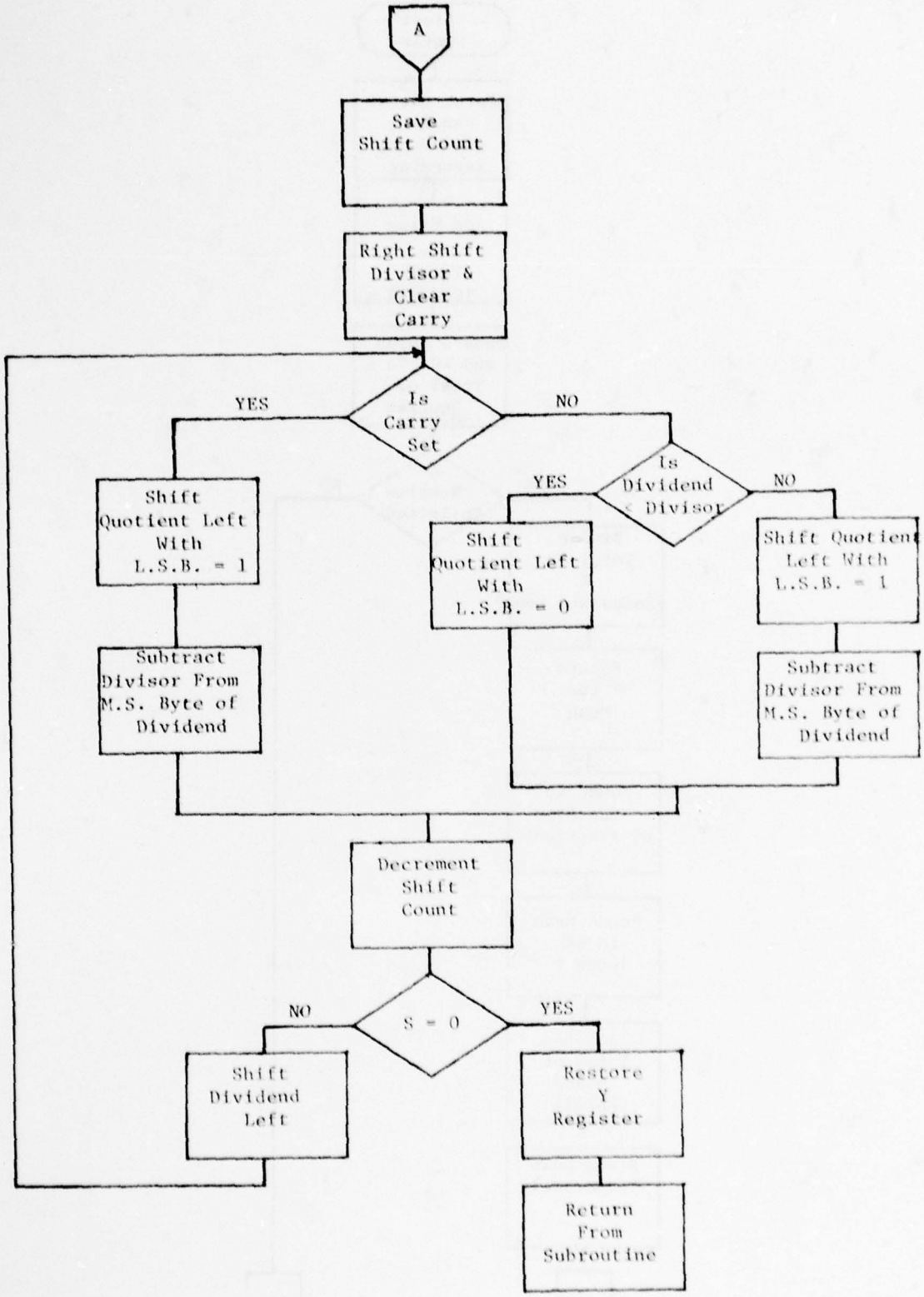


FIGURE 2.6: M-ATEC II DIVIDE SUBROUTINE (CONT)

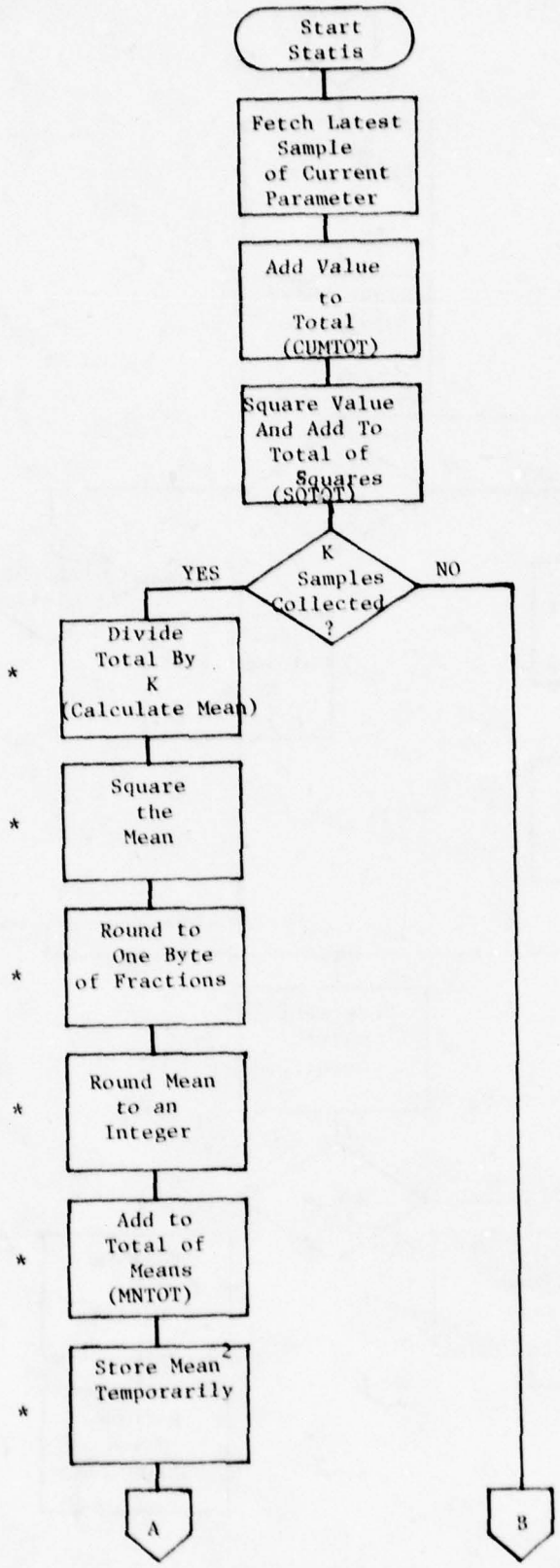


FIGURE 2.7: ATEC II STATISTICS SUBROUTINE FLOW CHART

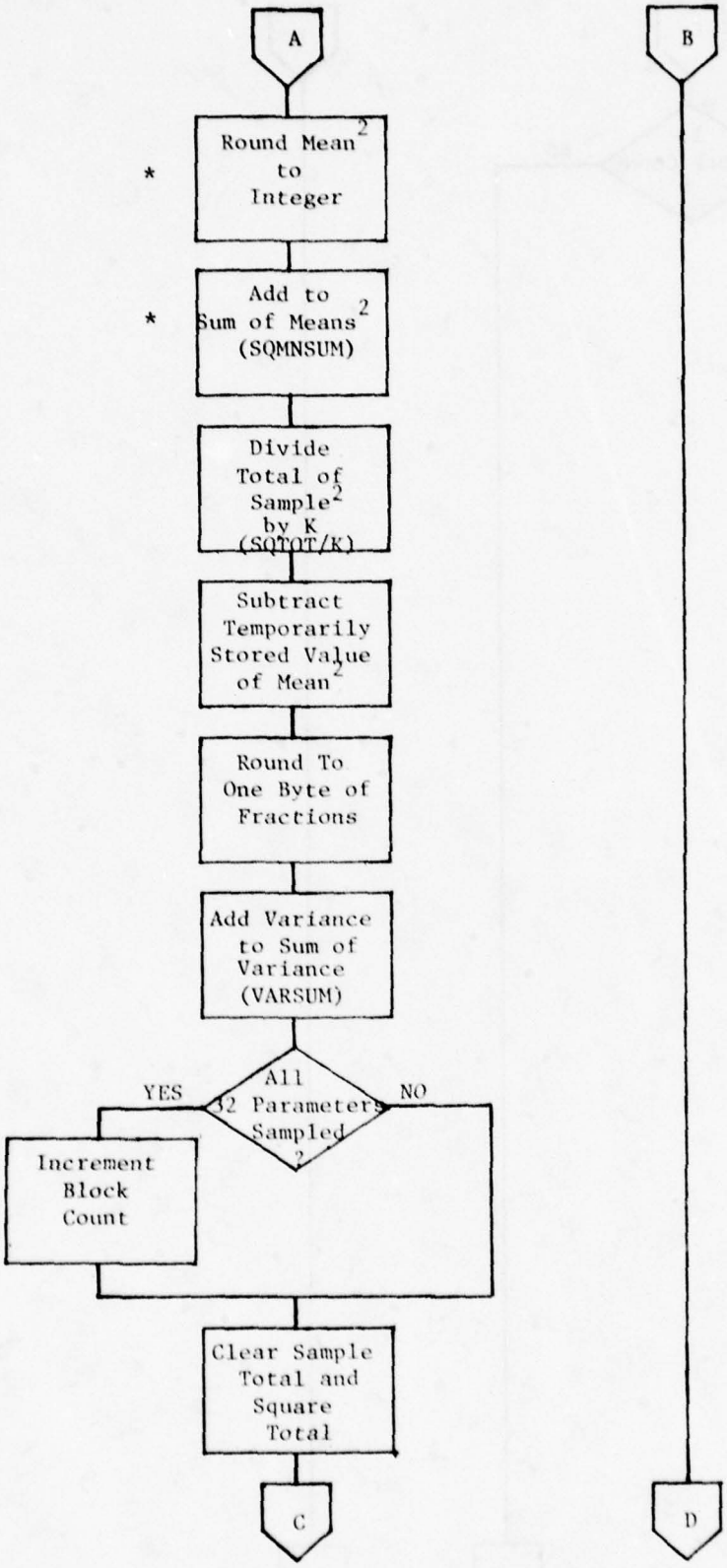


FIGURE 2.7: STATISTICS SUBROUTINE (CONT)

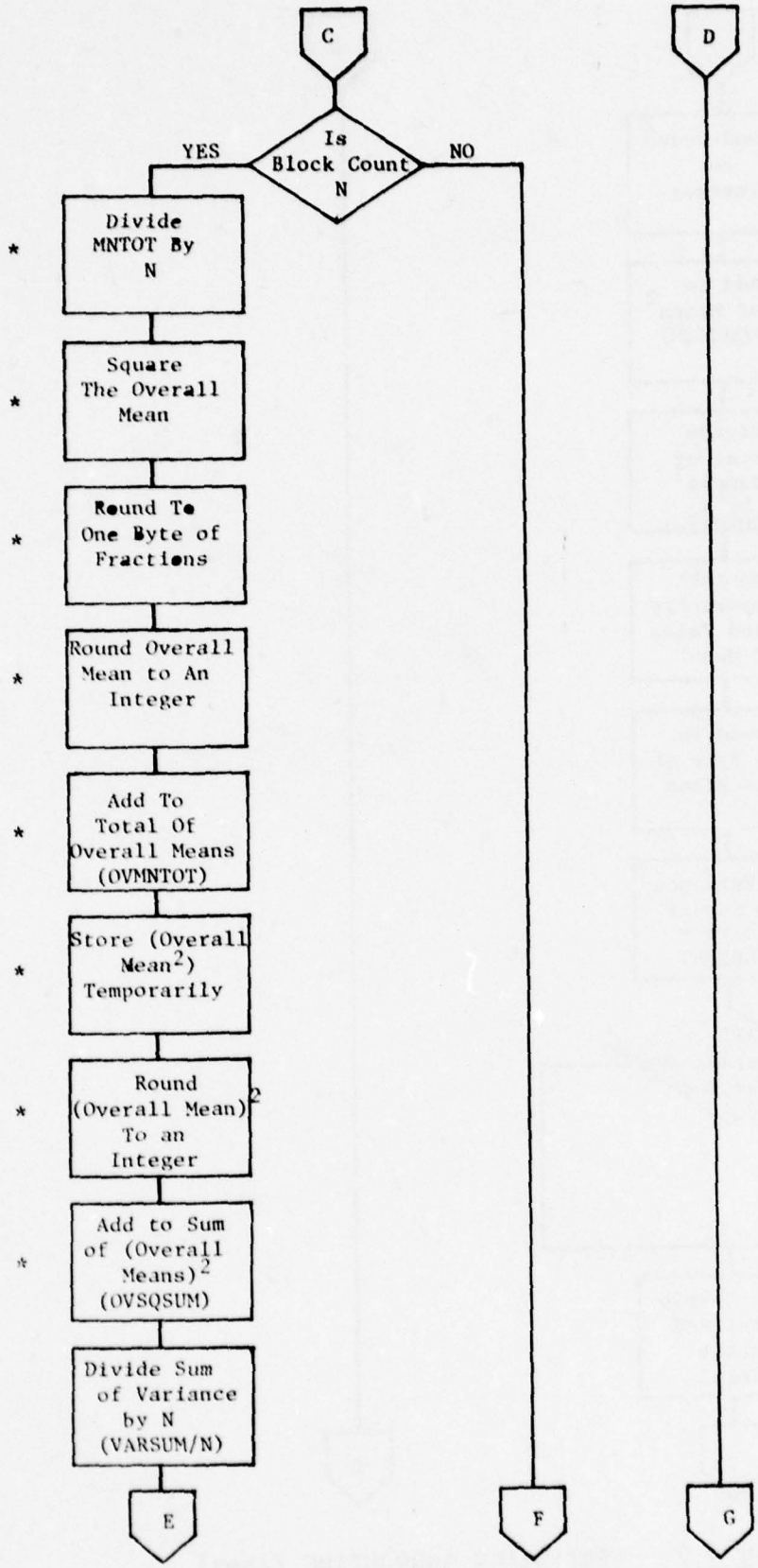


FIGURE 2.7: STATISTICS SUBROUTINE (CONT)

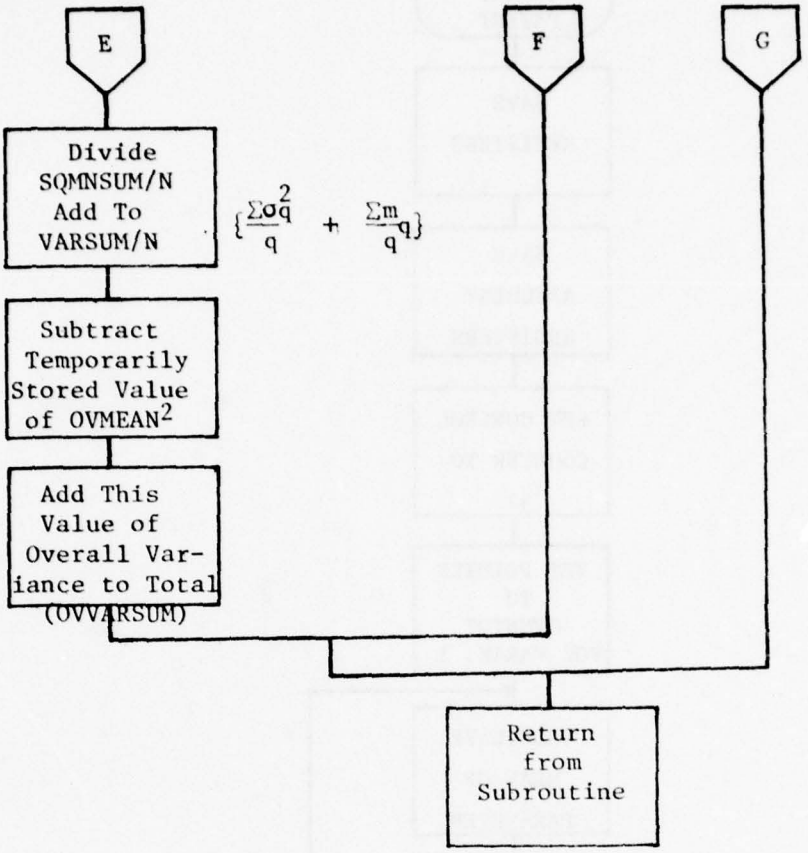


FIGURE 2.7: STATISTICS SUBROUTINE (CONT)

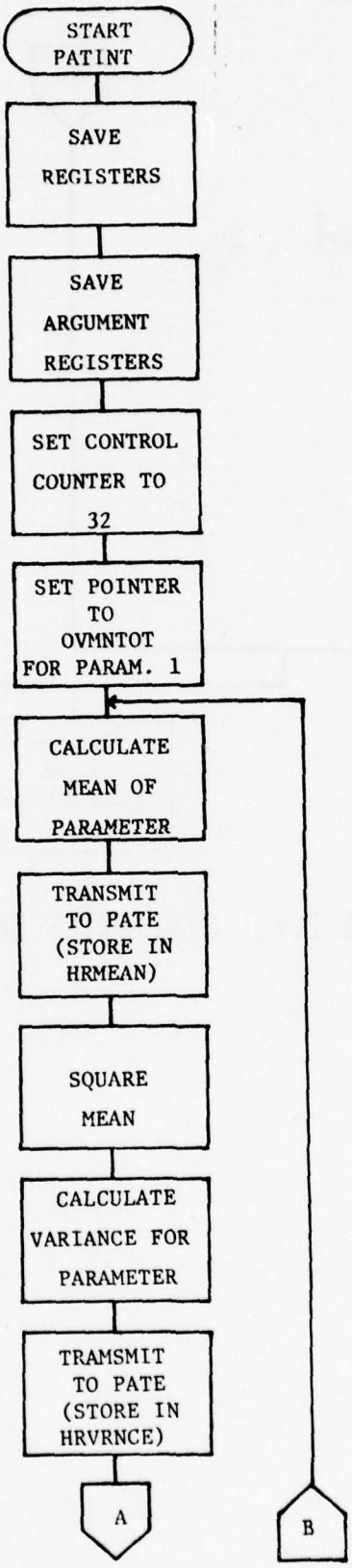


FIGURE 2.8: PATE INTERRUPT SERVICE ROUTINE

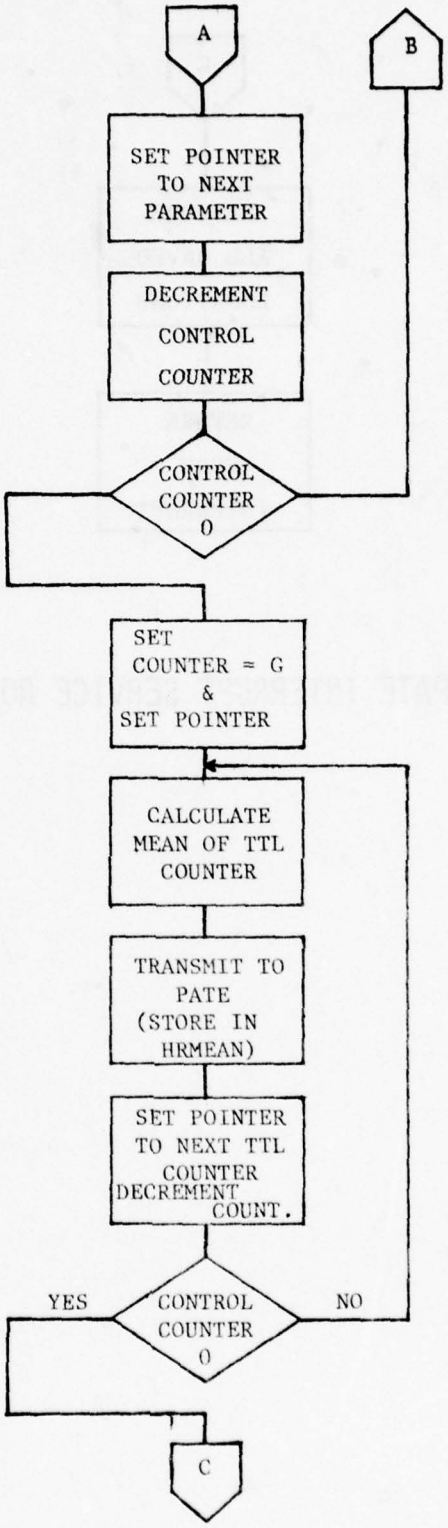


FIGURE 2.8: PATE INTERRUPT SERVICE ROUTINE (CONT'D)

FIGURE 2.8: PATE INTERRUPT SERVICE ROUTINE (CONT'D)

FIGURE 2.9: M-ATEC II STATISTICS ROUTINE

MEMORY MAP

PAGE ZERO

ADDRESS	ITEM
0004	ADD
0005	ADD+1
0006	ADD+2
0007	ADD+3
0008	AUG
0009	AUG+1
000A	AUG+2
000B	AUG+3
000C	MULP
000D	MULP+1
000F	MSIZE
00010	SIZE

PAGES 2-5

ADDRESS	ITEM
0200	CONSTANTS AND ACCUMULATED VALUES FOR PARAMETER 1
021B	CONSTANTS ETC, FOR MEASURED PARAMETER 2
0236	CONSTANTS ETC. FOR MEASURED PARAMETER 3
0545 0560	CONSTANTS ETC. FOR MEASURED PARAMETER 32

ADDRESS	ITEM
021F	CUMTOT
0220	CUMTOT+1
0221	SQTOT
0222	SQTOT+1
0223	SQTOT+2
0224	MNTOT
0225	MNTOT+1
0226	SQMNSUM
0227	SQMNSUM+1
0228	SQMNSUM+2
0229	OVMNTOT
022A	OVMNTOT+1
022B	OVSQSUM
022C	OVSQSUM+1
022D	OVSQSUM+2
022E	VARSUM
022F	VARSUM+1
0230	OVVARSUM
0231	OVVARSUM+1
0232	OVCNT

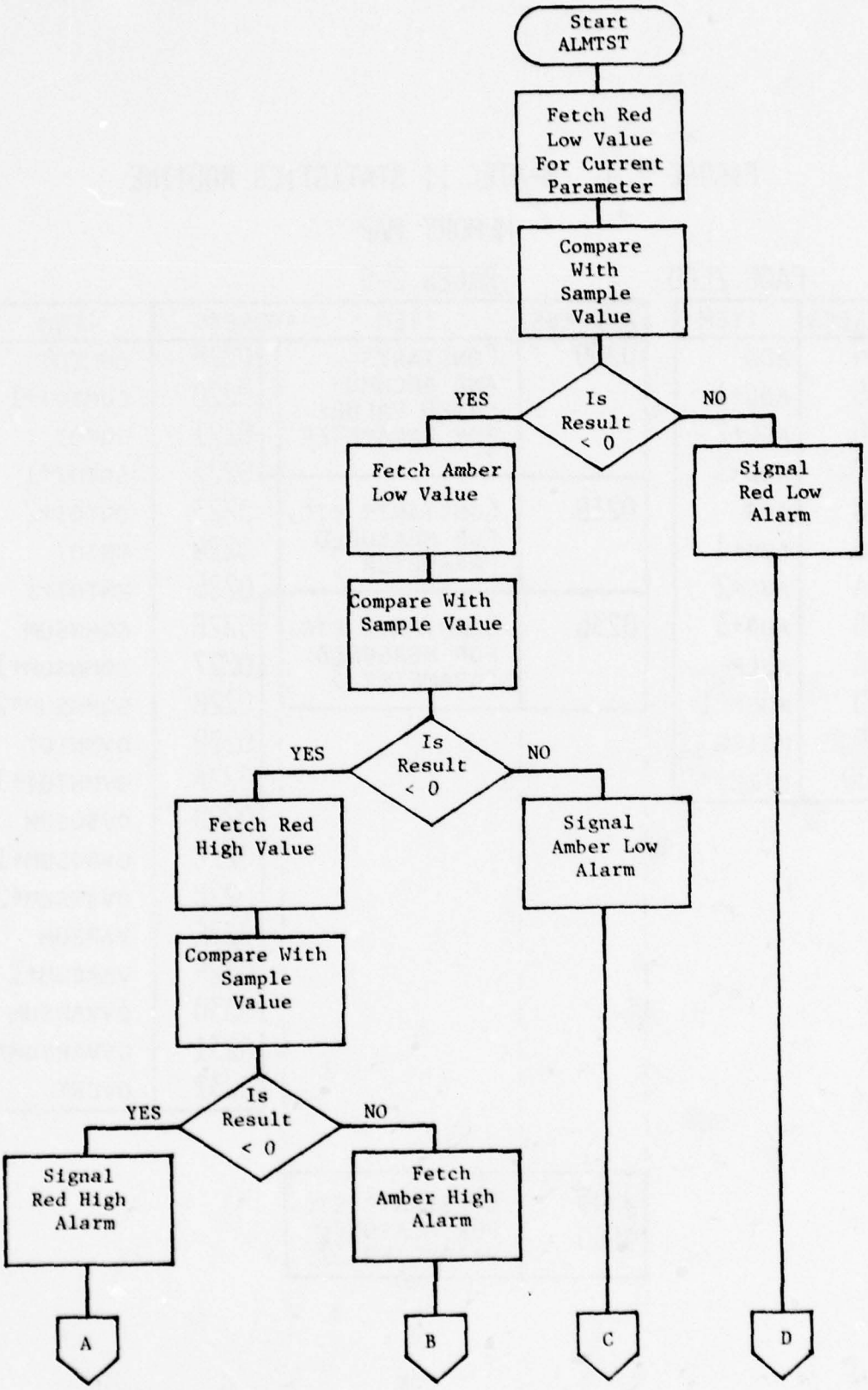


FIGURE 2.10: M-ATEC II ALARM LEVEL TEST PROGRAM

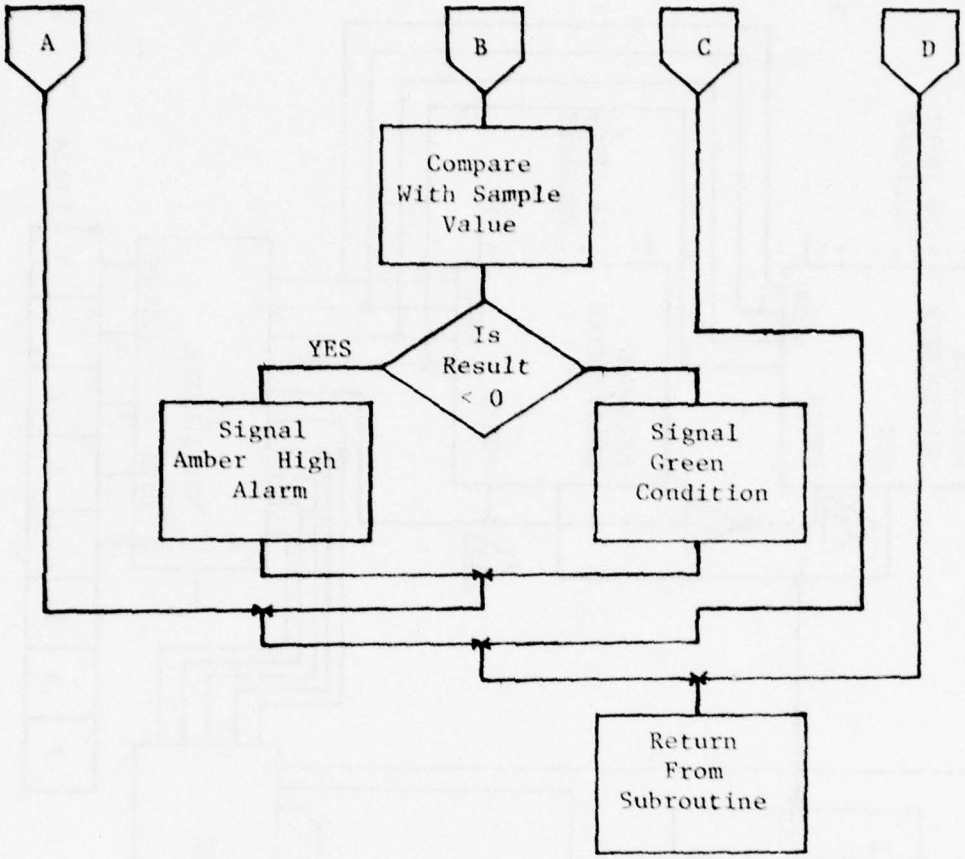


FIGURE 2.10: M-ATEC II ALARM LEVEL TEST PROGRAM (CONT)

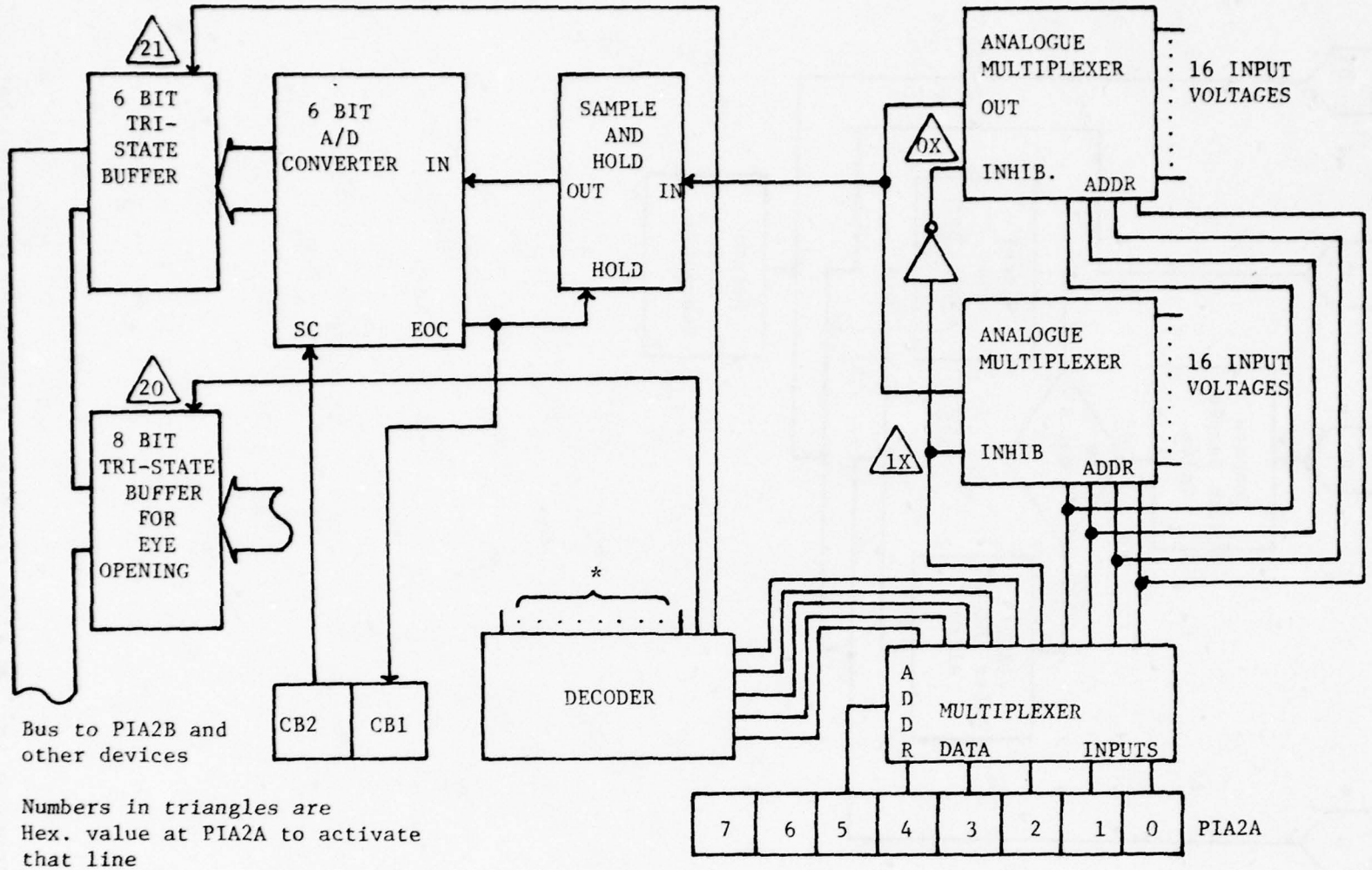


FIGURE 2.11: M-ATEC II ANALOGUE VOLTAGE SCAN INTERFACE

FIGURE 2.12: M-ATEC II ANALOGUE VOLTAGE SCAN ROUTINE (ANVSCN) MEMORY MAP

PAGE ZERO

ADDRESS	ITEM
0000	TEMP
0001	TEMP1
0002	TEMP2
0004	ADD
0005	ADD+1
0006	ADD+2
0007	ADD+3
0008	AUG
0009	AUG+1
000A	AUG+2
000B	AUG+3
000C	MULP
000D	MULP+1
000E	MULP+2
000F	MSIZE
0010	SIZE
0011	POINTL1
0012	POINTH1
0017	SAMCNT
0018	BLKCNT
0019	ANCTR

PAGES 2-5

ADDRESS	ITEM
0200	CONSTANTS AND ACCUMULATED VALUES FOR PARAMETER 1
021B	CONSTANTS ETC. FOR MEASURED PARAMETER 2
0236	CONSTANTS ETC. FOR MEASURED PARAMETER 3
0545 0560	CONSTANTS ETC. FOR MEASURED PARAMETER 32

ADDRESS	ITEM
021B	RED LOW LEVEL
021C	AMBER LOW LEVEL
021D	RED HIGH LEVEL
021E	AMBER HIGH LEVEL
021F	CUMTOT
0220	CUMTOT+1
0221	SQTOT
0222	SQTOT+1
0223	SQTOT+2
0224	MNTOT
0225	MNTOT+1
0226	SQMNSUM
0227	SQMNSUM+1
0228	SQMNSUM+2
0229	OVMNTOT
022A	OVMNTOT
022B	OVSQSUM
022C	OVSQSUM+1
022D	OVSQSUM+2
022E	VARSUM
022F	VARSUM+1
0230	OVVARSUM
0231	OVVARSUM+1
0232	OVCNT
0233	ALRMST
0234	HRVARNCE
0235	HRMEAN

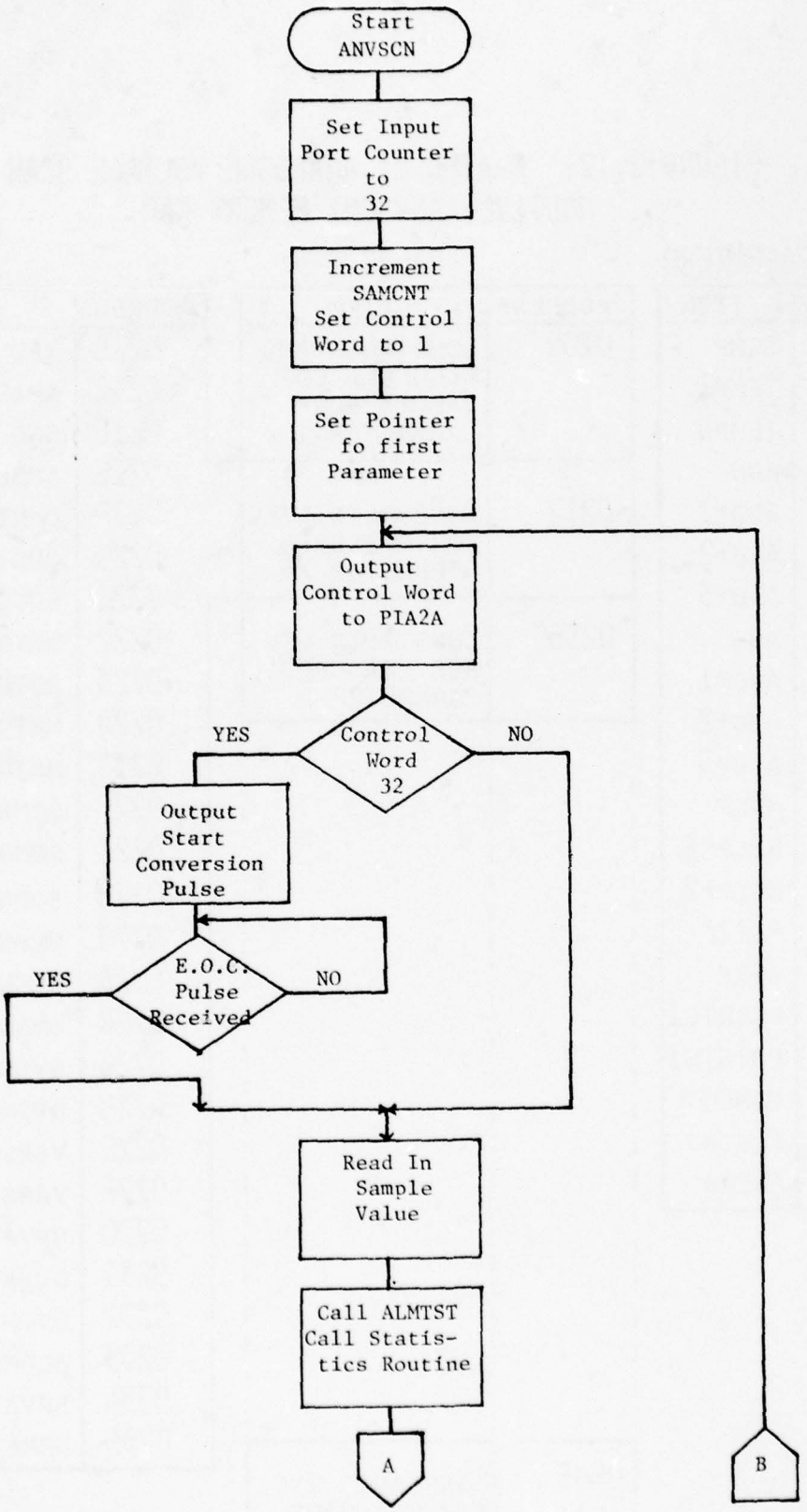


FIGURE 2.13: M-ATEC II ANALOGUE VOLTAGE SCAN ROUTINE

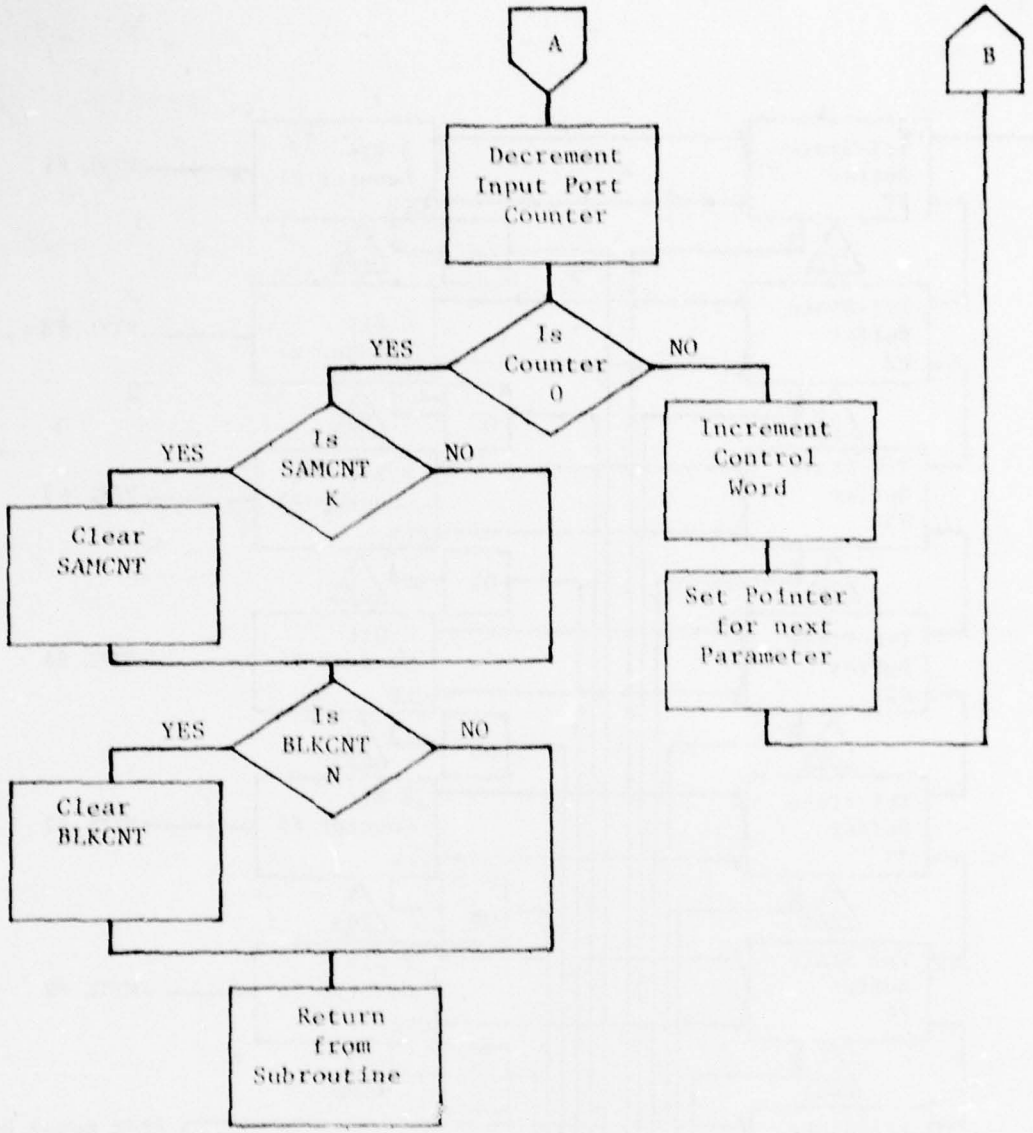


FIGURE 2.13: M-ATEC II ANALOGUE VOLTAGE SCAN ROUTINE (CONT)

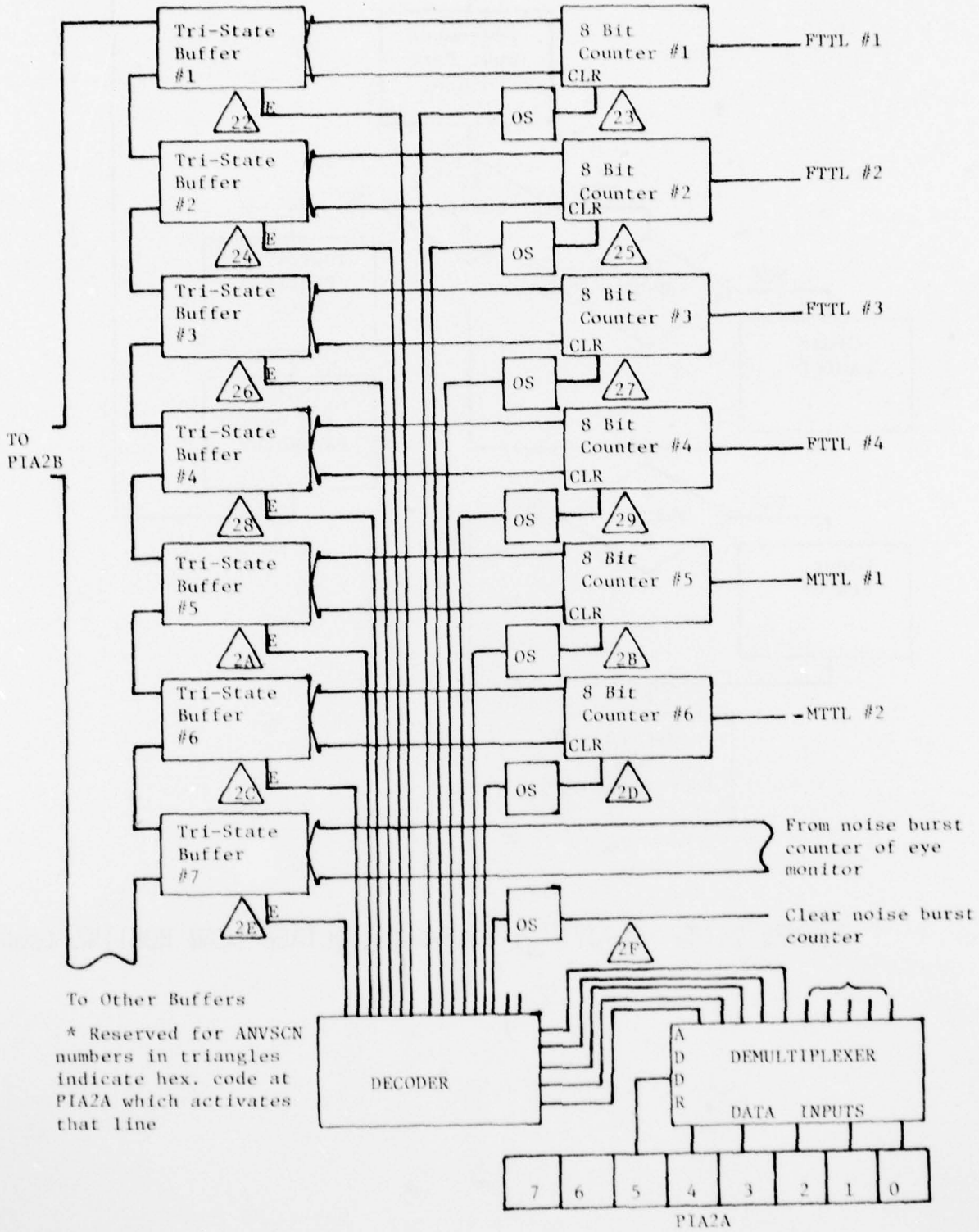


FIGURE 2.14: TTL COUNTER INTERFACE

FIGURE 2.15: M-ATEC II TTL INTERRUPT ROUTINE MEMORY MAP

PAGE ZERO

ADDRESS	ITEM
0003	TEMP4
0013	POINTL2
0014	POINTH2
001A	TSBN
001B	CNTN

PAGE 5-6

ADDRESS	ITEM
0561	ACCUMULATED VALUES FOR FTTL COUNTER 1
056F	ACCUMULATED VALUES FOR FTTL COUNTER 2
057D	ACCUMULATED VALUES FOR FTTL COUNTER 3
058B	ACCUMULATED VALUES FOR FTTL COUNTER 4
0599	ACCUMULATED VALUES FOR MTTL COUNTER 1
05A7	ACCUMULATED VALUES FOR MTTL COUNTER 2
05B5 - 05C2	ACCUMULATED VALUES FOR NOISE BURST COUNTER

ADDRESS	ITEM
056F	CUMTOT
0570	CUMTOT+1
0571	SAMCNT
0572	MNSUM
0573	MNSUM+1
0574	BLKCNT
0575	OVMNSUM
0576	OVMNSUM+1
0577	OVCNT
0578	NTSBN
0579	NCNTN
057A	NPNTL
057B	NPNTH
057C	HRMEAN

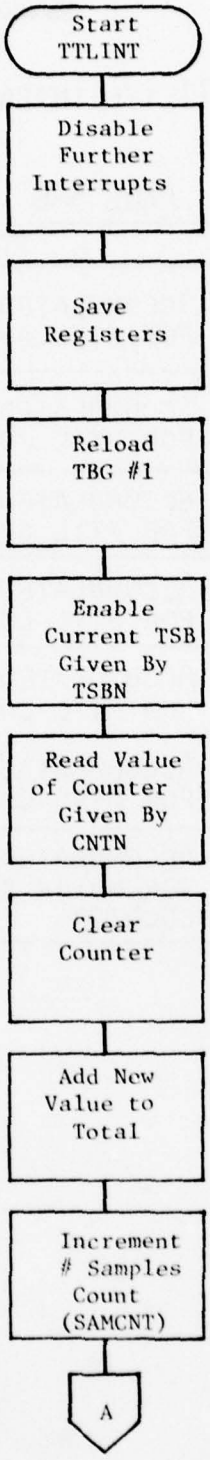


FIGURE 2.16: M-ATEC II TTL COUNTER SERVICE ROUTINE

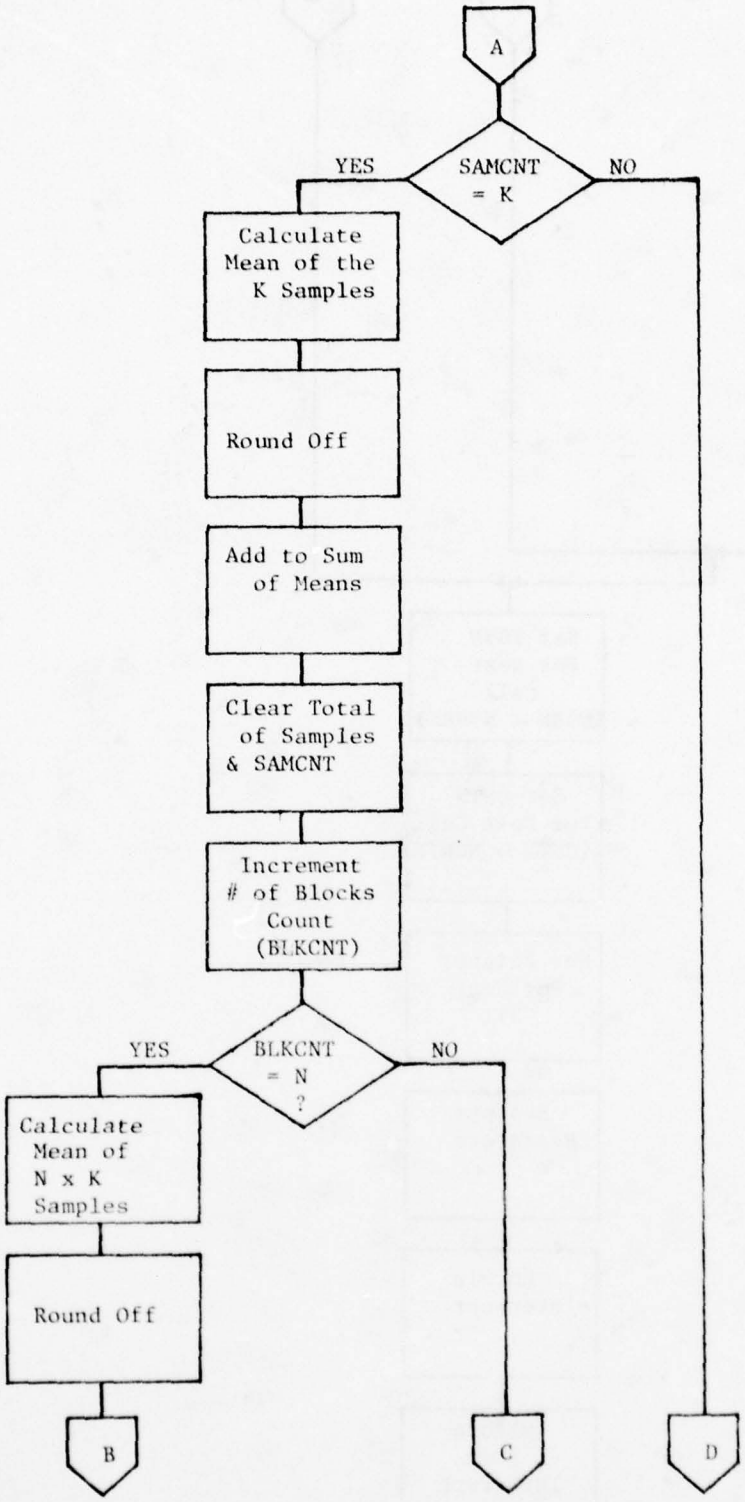


FIGURE 2.16: M-ATEC II TTL COUNTER SERVICE ROUTINE (CONT)

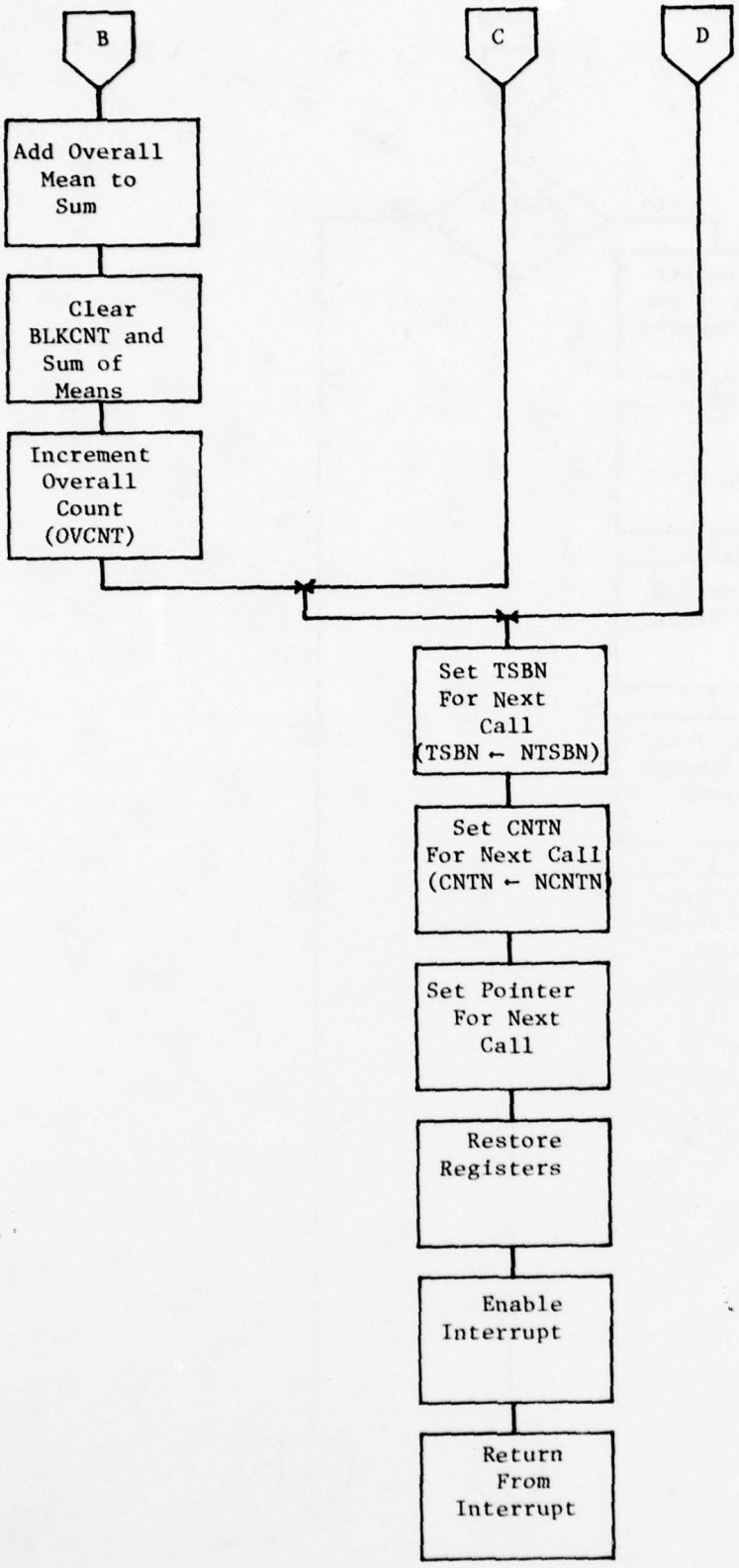
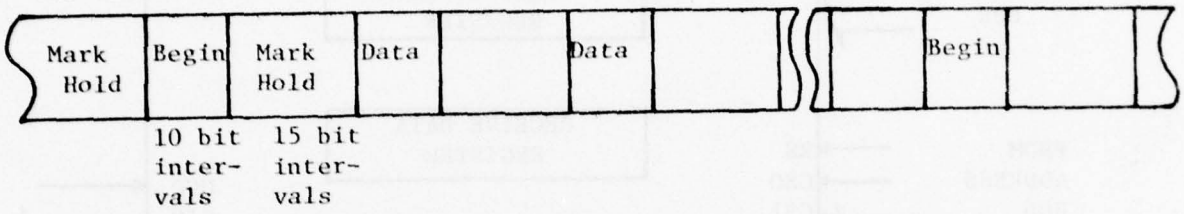


FIGURE 2.16: M-ATEC II COUNTER SERVICE ROUTINE (CONT)

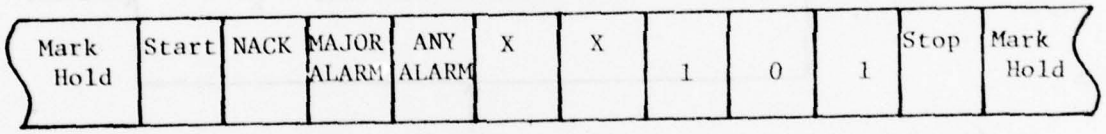
FIGURE 3.1: M-ATEC II ALARM DISPLAY MESSAGE FORMAT

3.1.1: Message Organization

The data rate is 75 bits.sec^{-1}

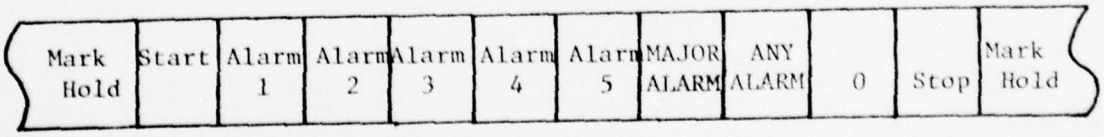


3.1.2 Begin Character



x - don't care

3.1.3 Data Character



Each data character contains the data for five alarms

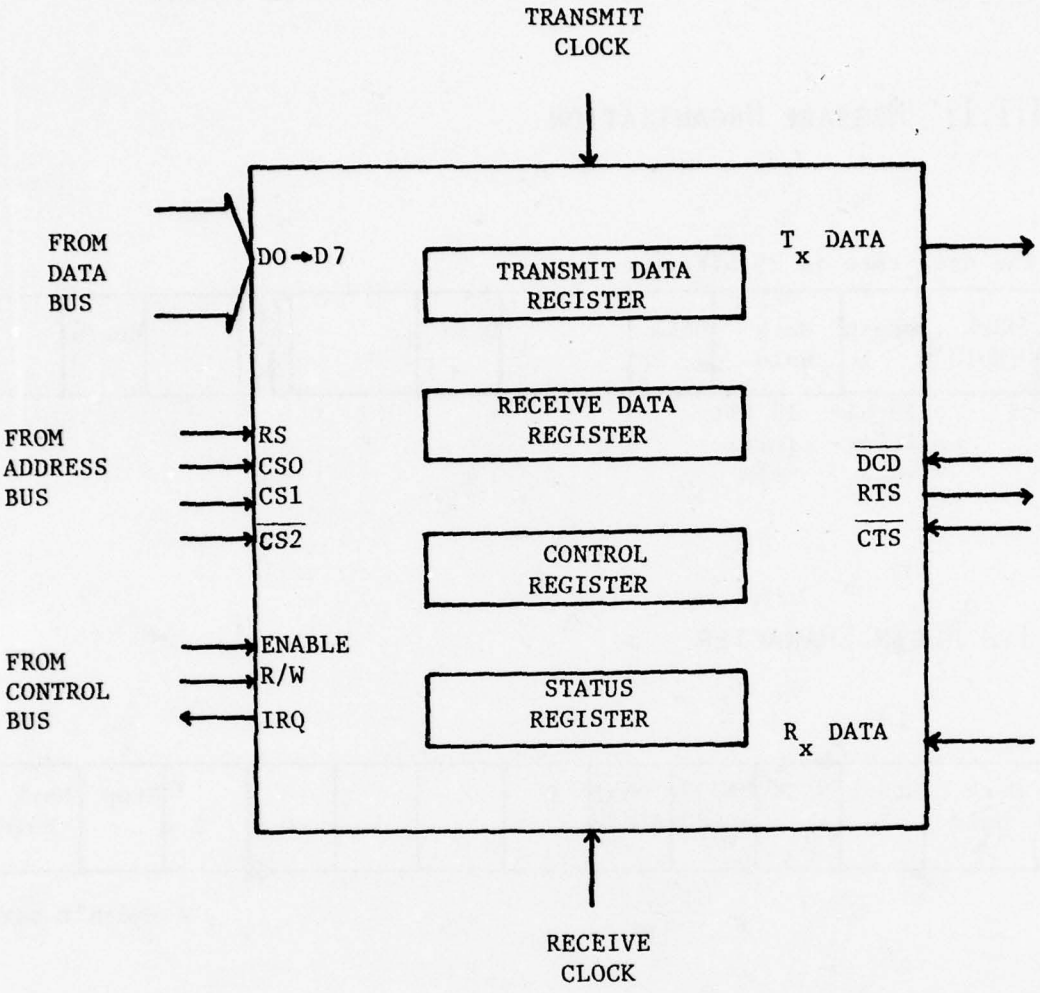


FIGURE 3.2: ASYNCHRONOUS COMMUNICATIONS INTERFACE ADAPTER

FIGURE 3.3: M-ATEC II ALARM SCANNER TRANSMISSION SEQUENCE

EVENT	TIME (μs)
RECEIVE INTERRUPT FROM TIME BASE GENERATOR. COMMAND ACIA TO STOP TRANSMITTING BLANKS. LOAD ACIA OUTPUT REGISTER WITH NEXT DATA OR BEGIN CHARACTER AS APPROPRIATE.	0
ACIA TRANSMITS THE FIRST BIT OF THE NEW CHARACTER.	13,333
LAST BIT OF THE CHARACTER IS TRANSMITTED ACIA GENERATES INTERRUPT REQUESTING MORE DATA. PROCESSOR COMMANDS ACIA TO TRANSMIT BLANKS.	146,666
FIRST BLANK TRANSMITTED.	160,000
AFTER FIFTEEN BLANKS ARE TRANSMITTED ELAPSED TIME IS	333,333
TIME BASE GENERATOR GENERATES ANOTHER INTERRUPT	333,333

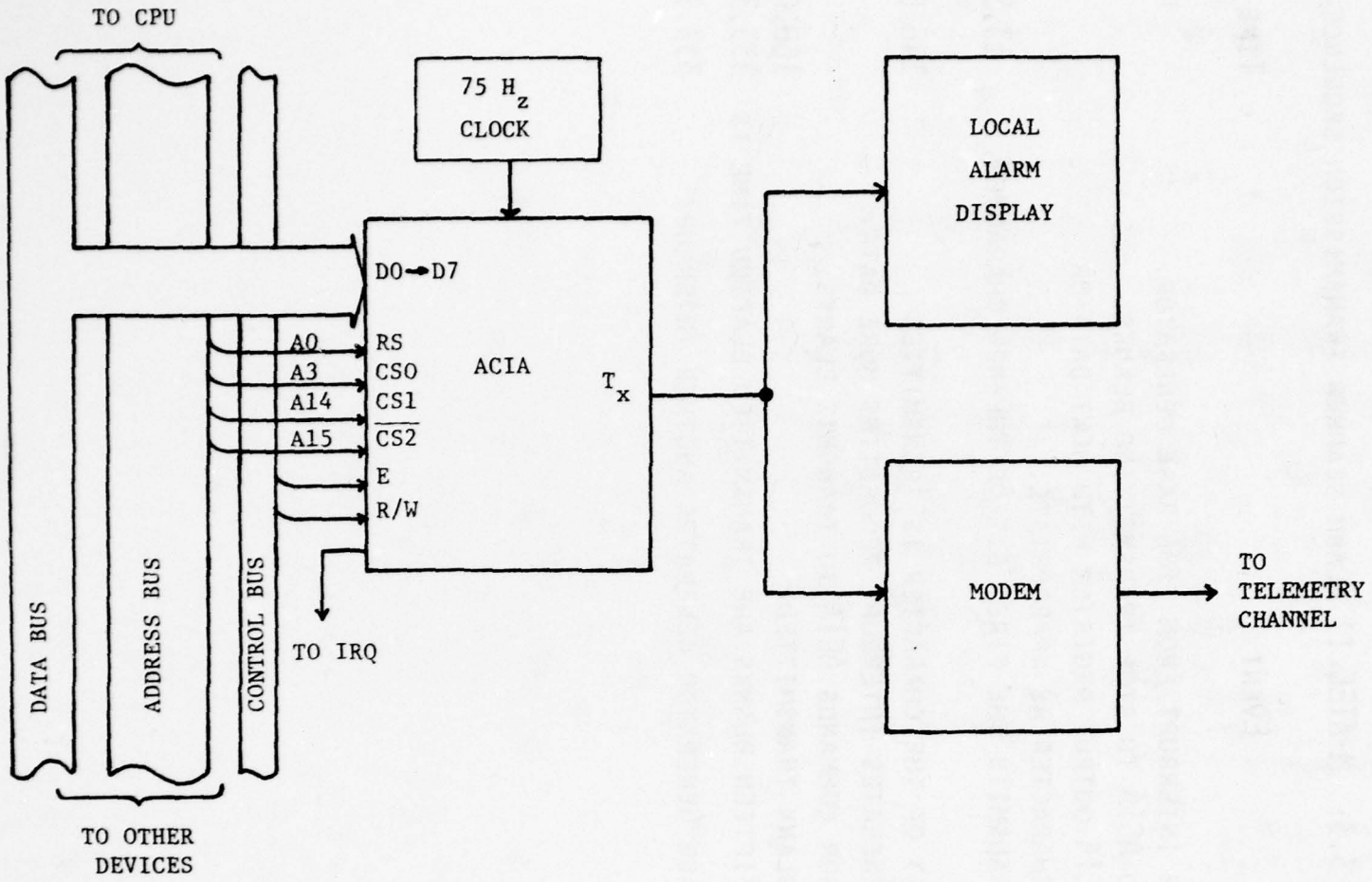


FIGURE 3.4: M-ATEC II LOCAL ALARM DISPLAY INTERFACE

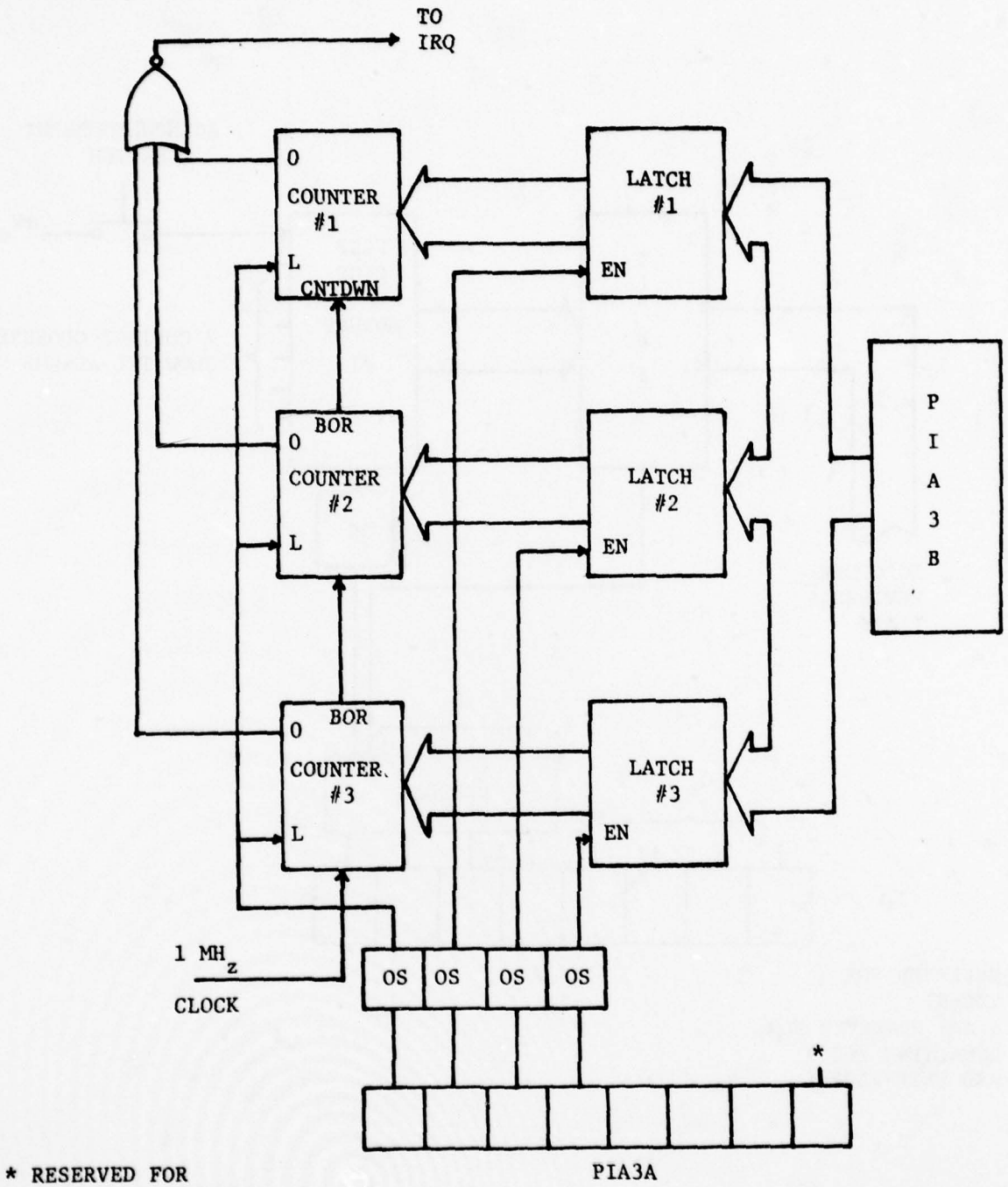


FIGURE 3.5: M-ATEC II TIME BASE GENERATOR 2

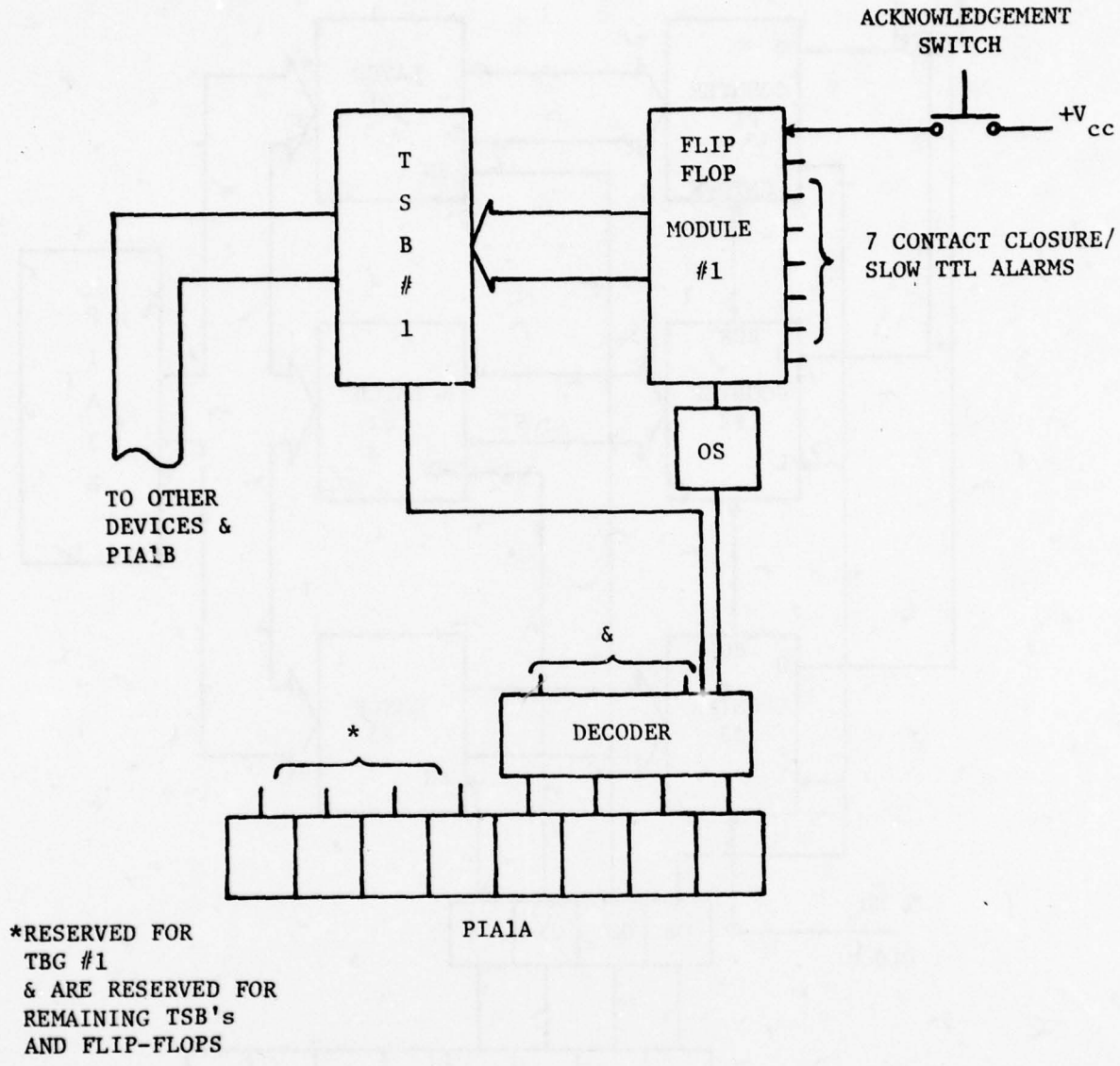


FIGURE 3.6: M-ATEC II ACKNOWLEDGEMENT SWITCH INTERFACE

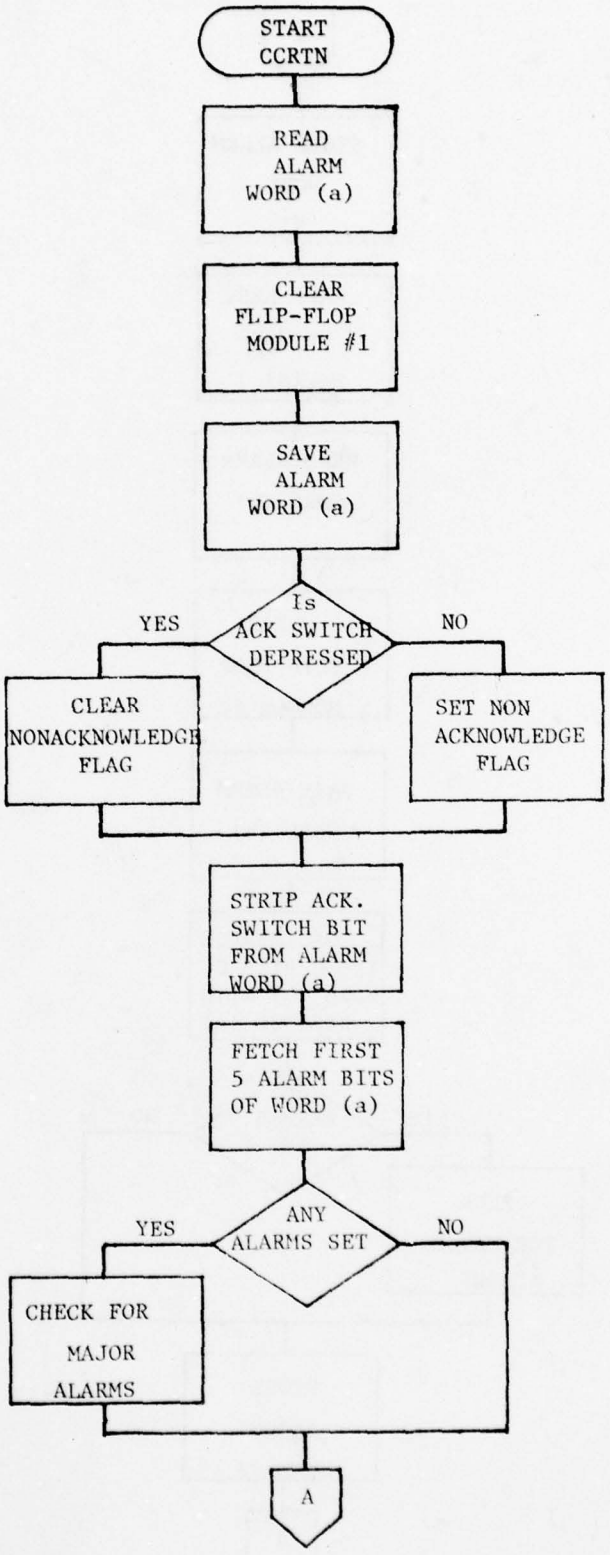


FIGURE 3.7: M-ATEC II ALARM SCAN ROUTINE

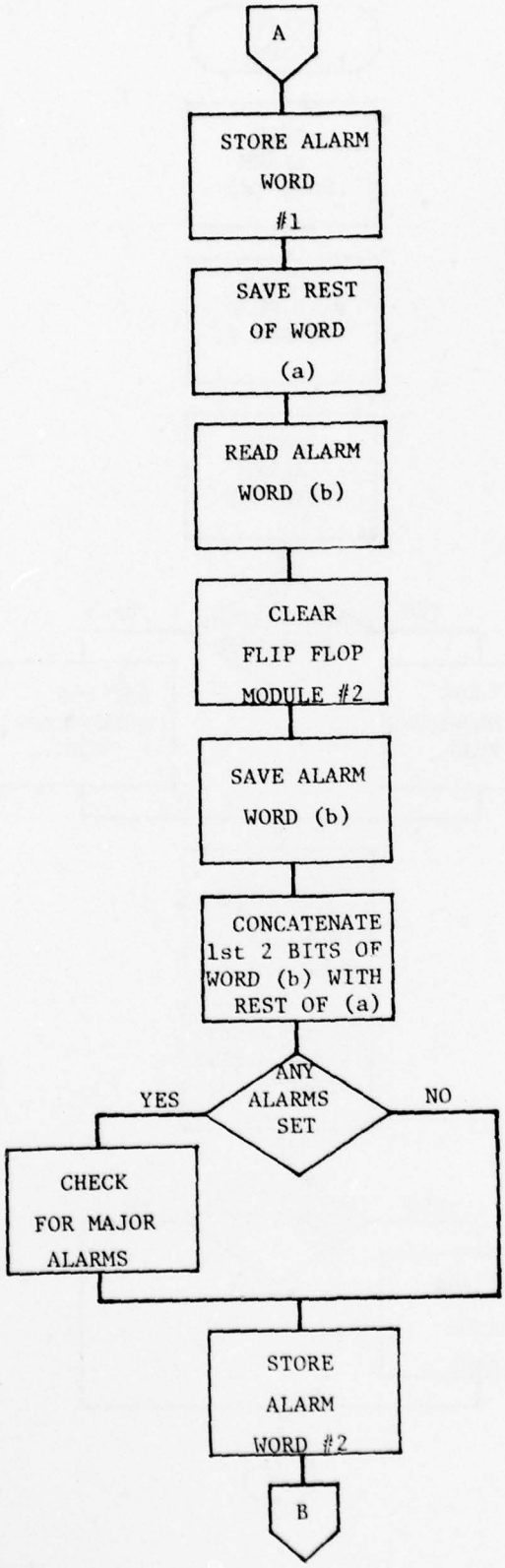
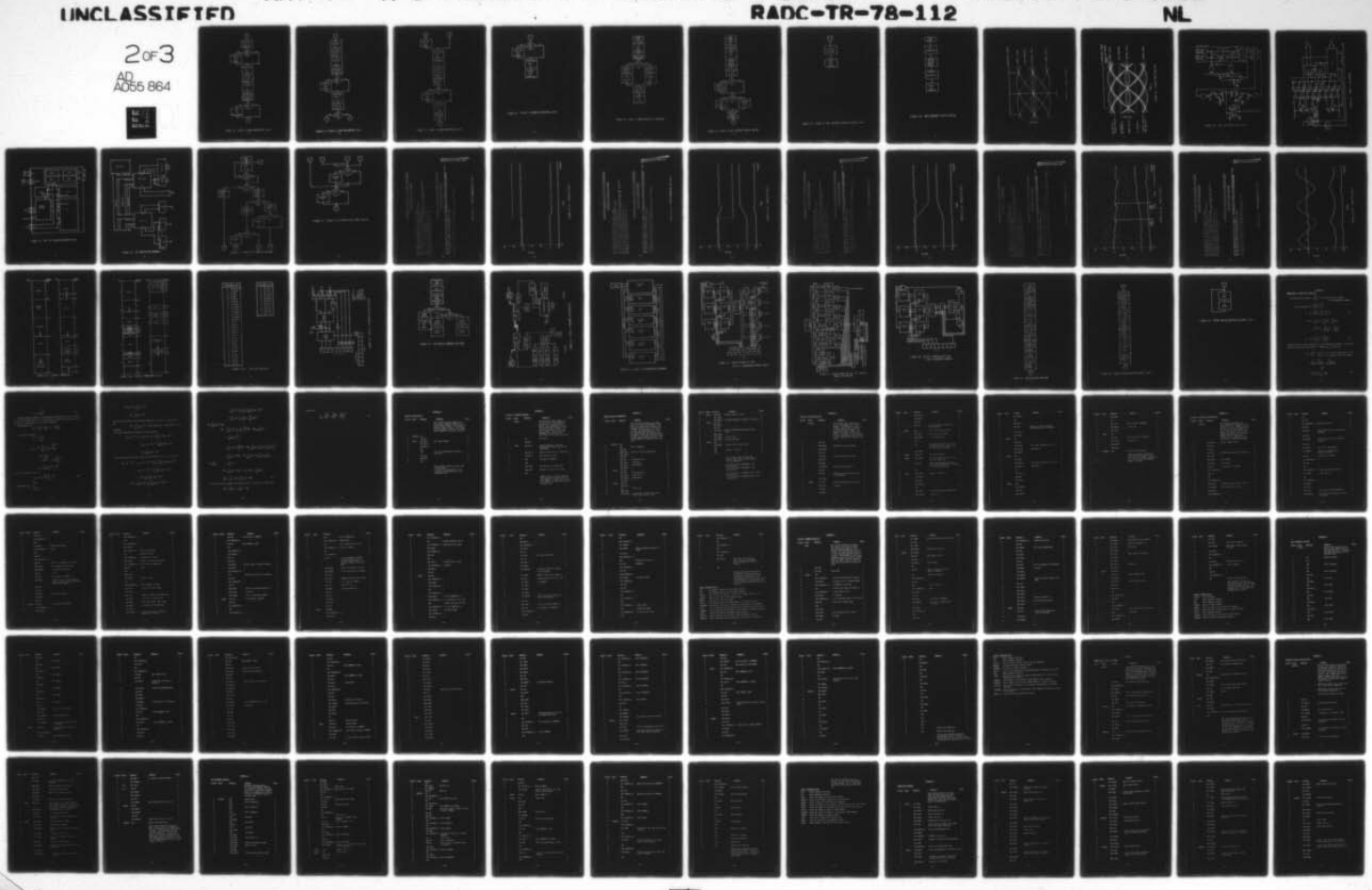


FIGURE 3.7: M-ATEC II ALARM SCAN ROUTINE (CONT)

AD-A055 864 PURDUE UNIV LAFAYETTE IND SCHOOL OF ELECTRICAL ENGI--ETC F/G 17/2
A MICROPROCESSOR BASED PERFORMANCE MONITOR FOR COMMUNICATION NE--ETC(U)
MAY 78 M S CALLOW, S C CRIST, B A BLACK F30602-75-C-0082
UNCLASSIFIED RADC-TR-78-112 NL

2 OF 3
AD
A055 864



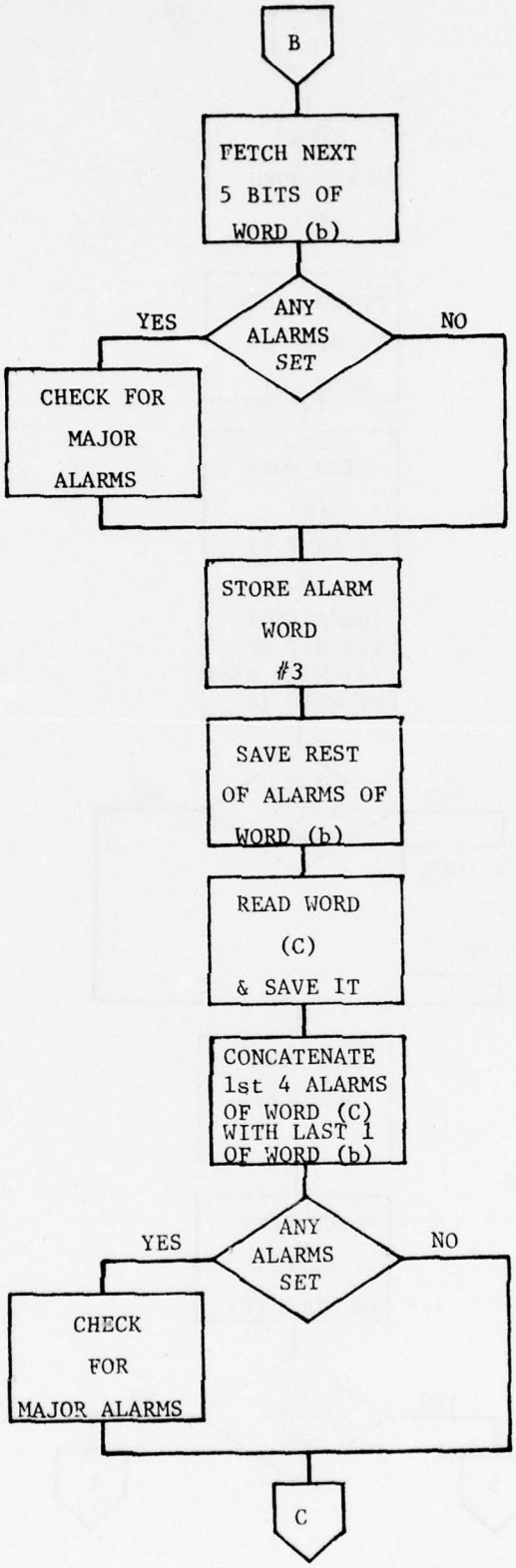


FIGURE 3.7: M-ATEC II ALARM SCAN ROUTINE (CONT)

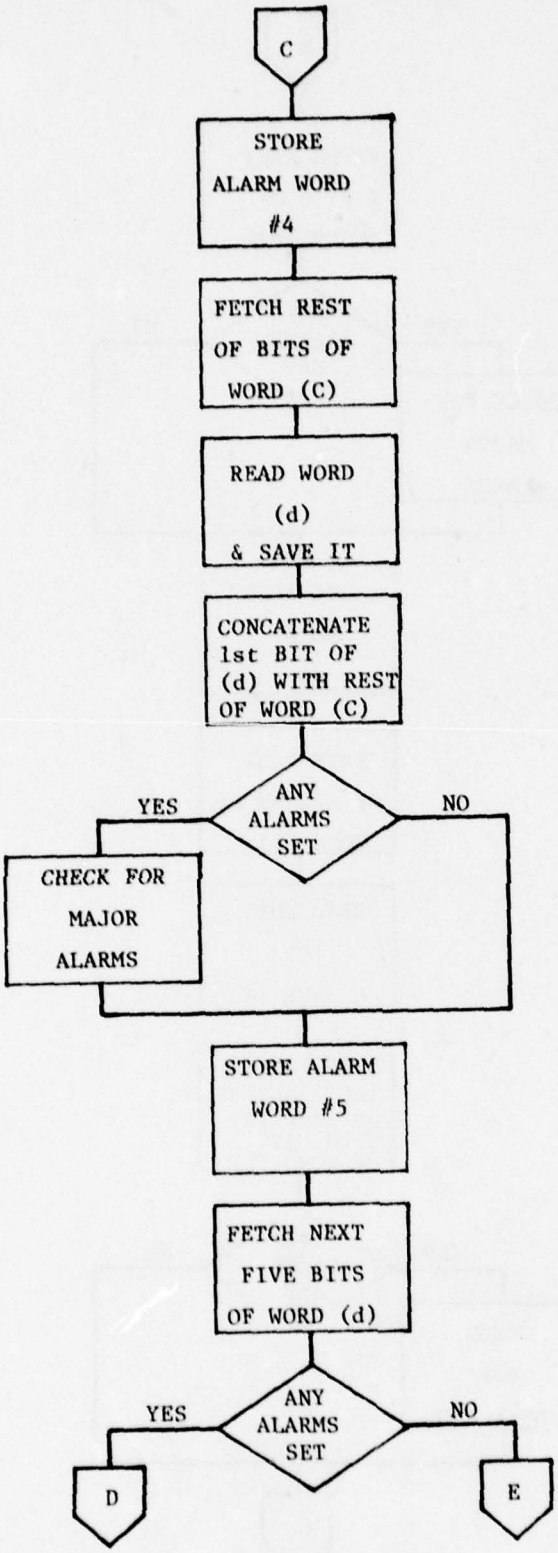


FIGURE 3.7: M-ATEC II ALARM SCAN ROUTINE (CONT)

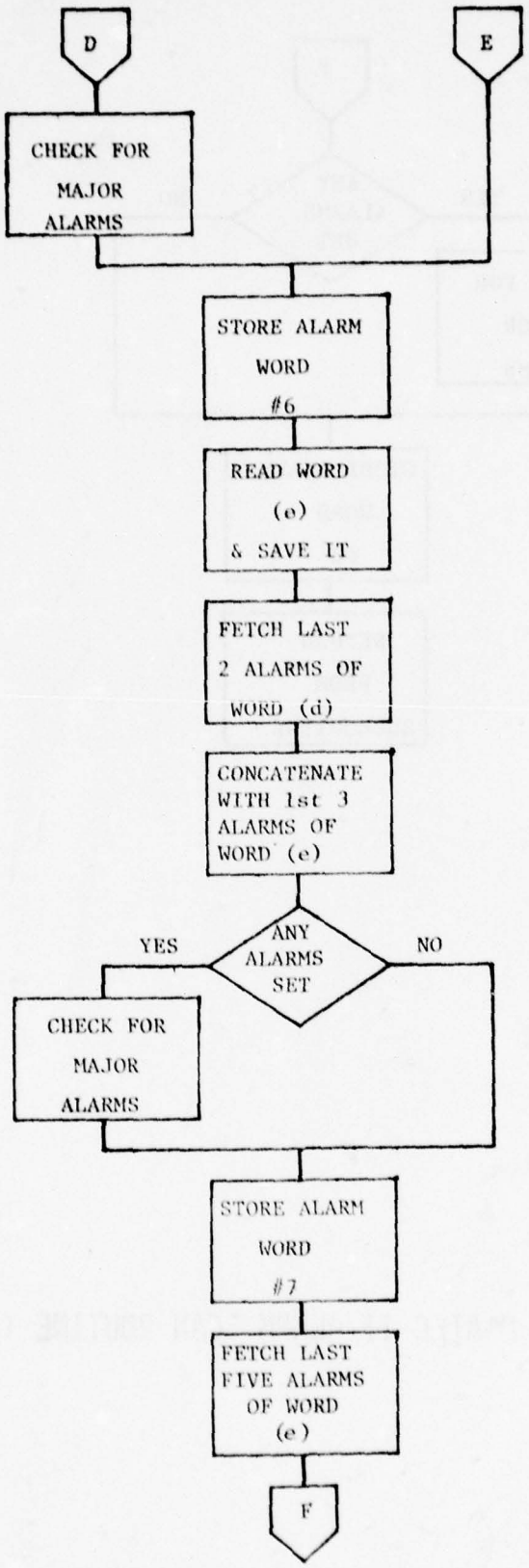


FIGURE 3.7: M-ATEC II ALARM SCAN ROUTINE (CONT)

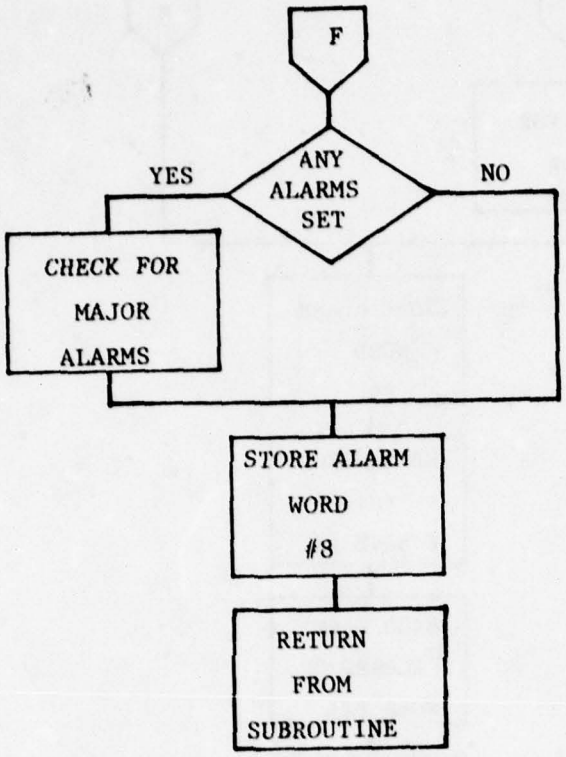


FIGURE 3.7: M-ATEC II ALARM SCAN ROUTINE (CONT)

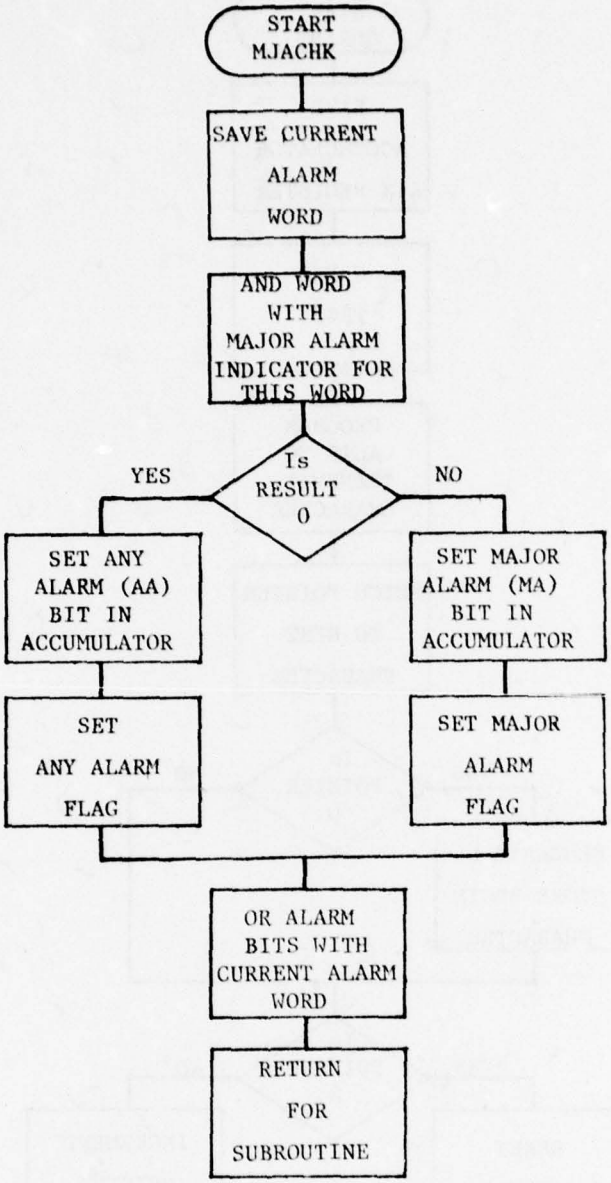


FIGURE 3.8: M-ATEC II MAJOR ALARM CHECK SUBROUTINE

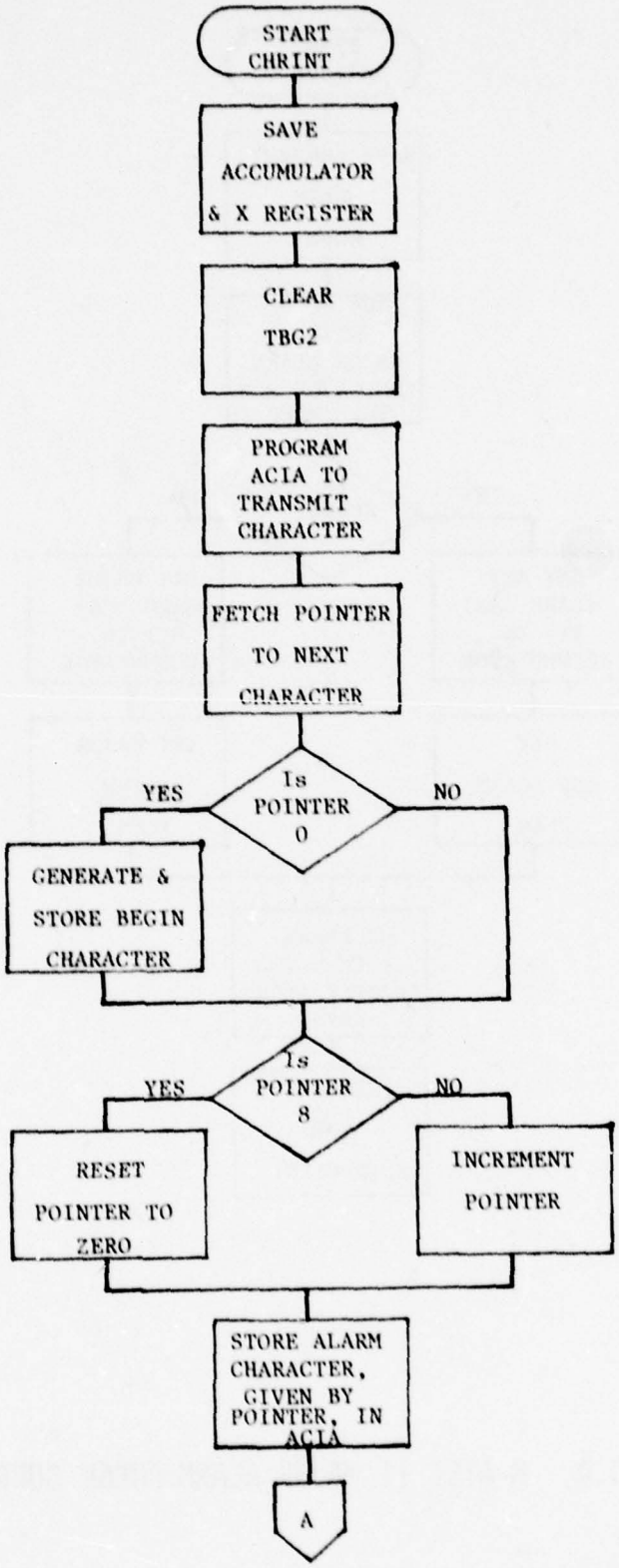


FIGURE 3.9: M-ATEC II TBG2 INTERRUPT SERVICE ROUTINE

FIGURE 3.9: M-ATEC II TBG2 INTERRUPT SERVICE ROUTINE (CONT)

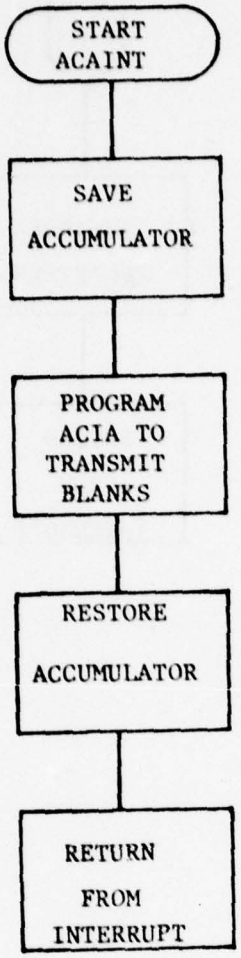


FIGURE 3.10: ACIA INTERRUPT SERVICE ROUTINE

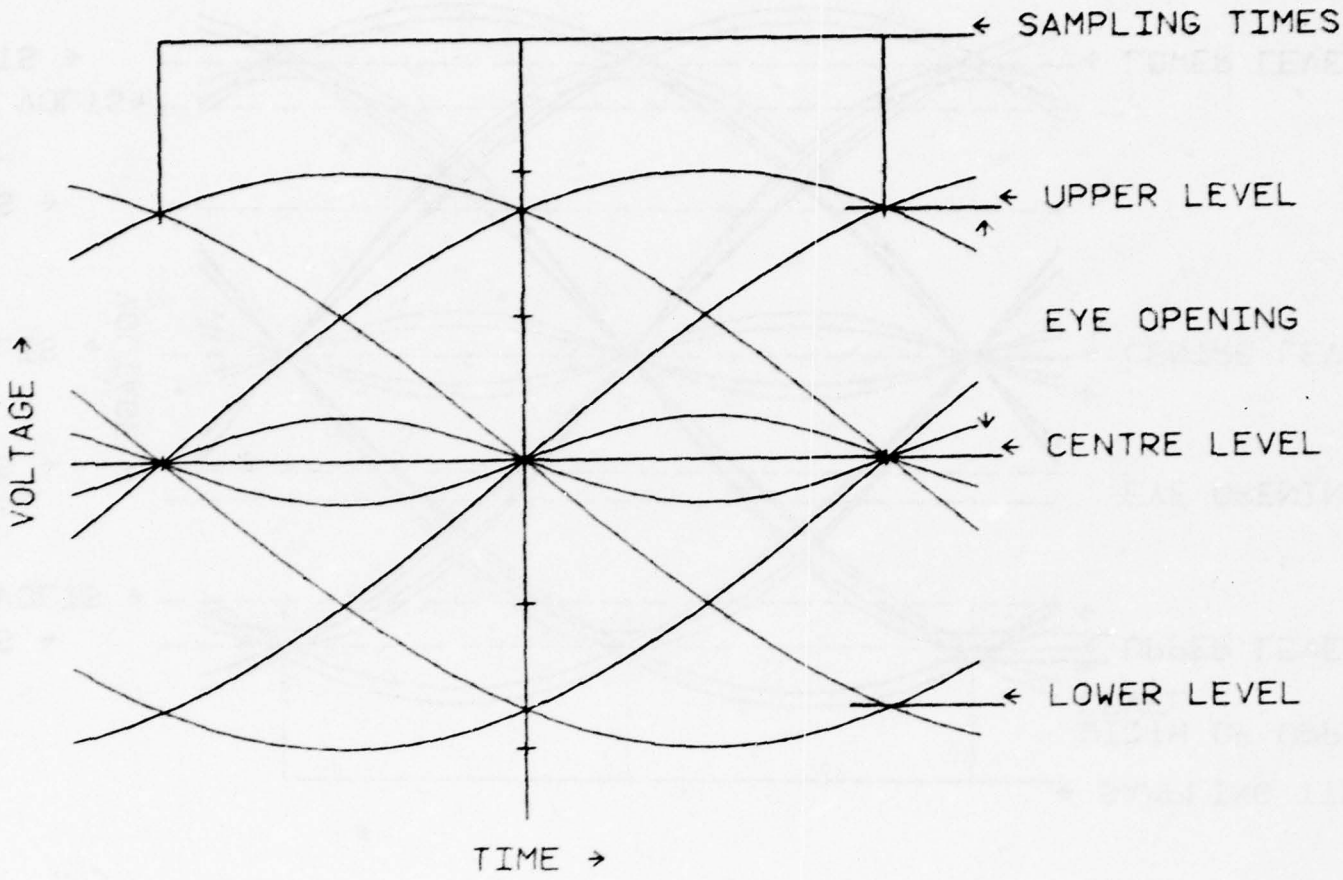


FIGURE 4.1: A THREE LEVEL EYE

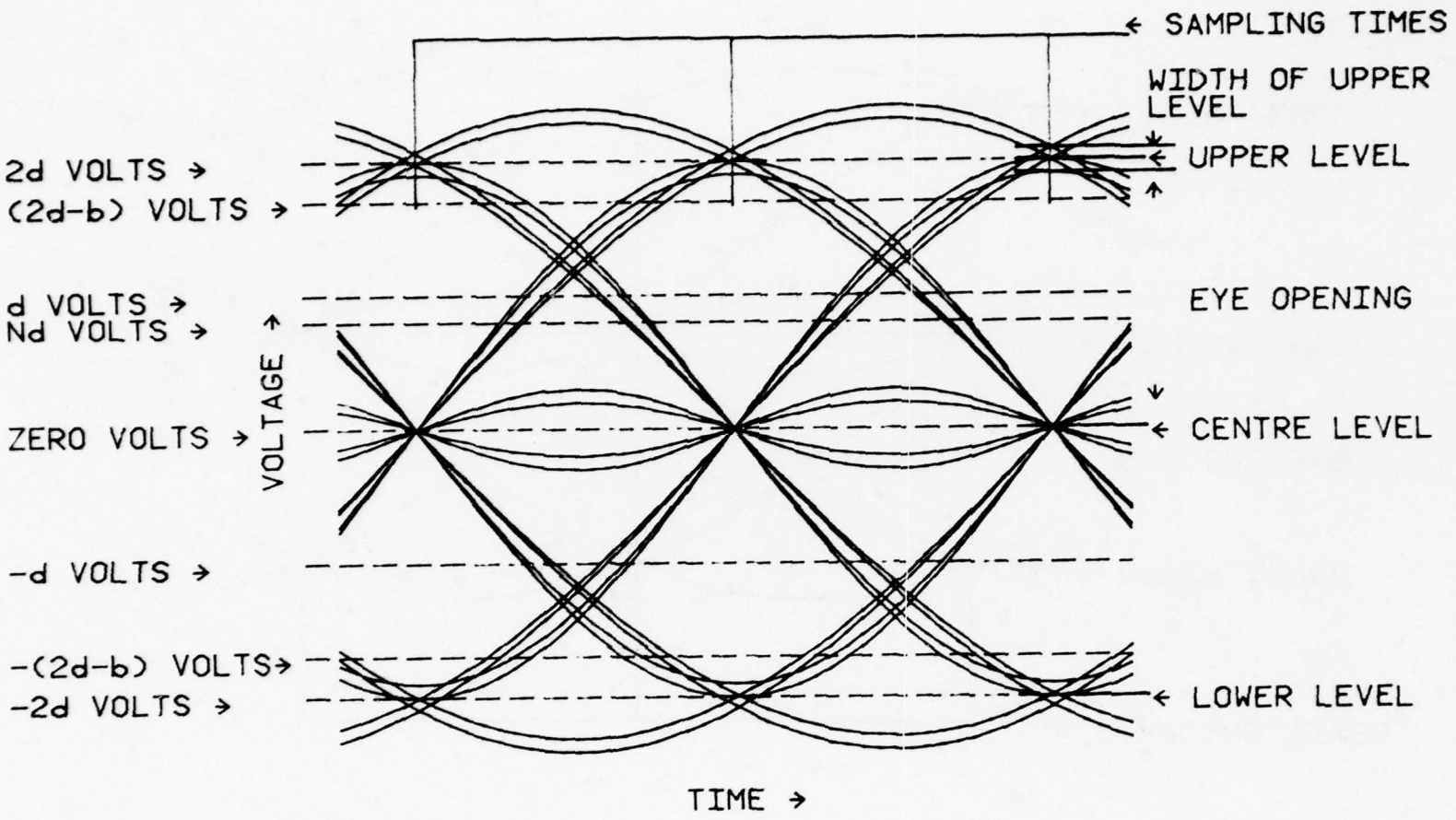


FIGURE 4.2: A NOISY THREE LEVEL EYE

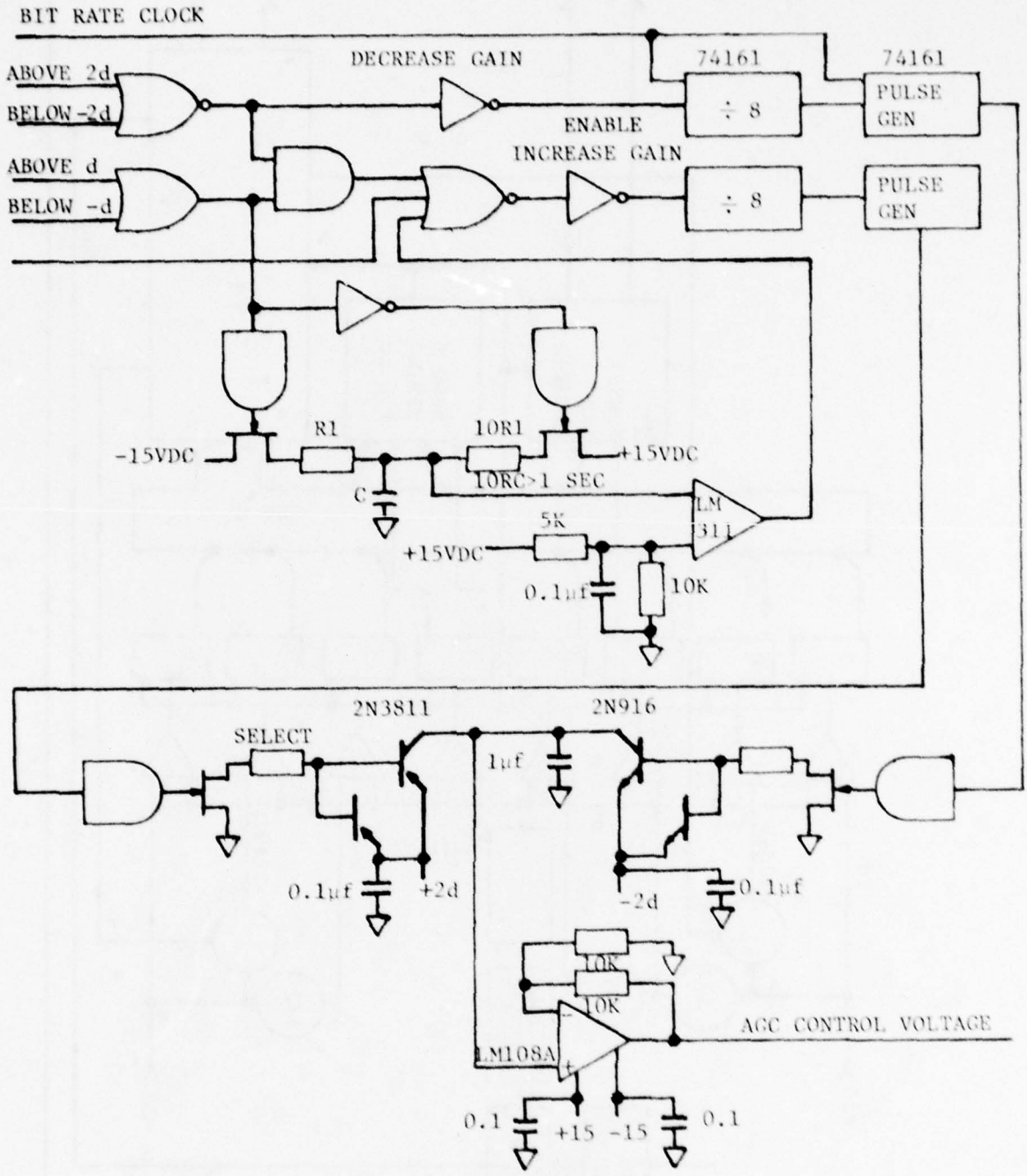


FIGURE 4.3: ATEC EYE MONITOR AGC CIRCUIT

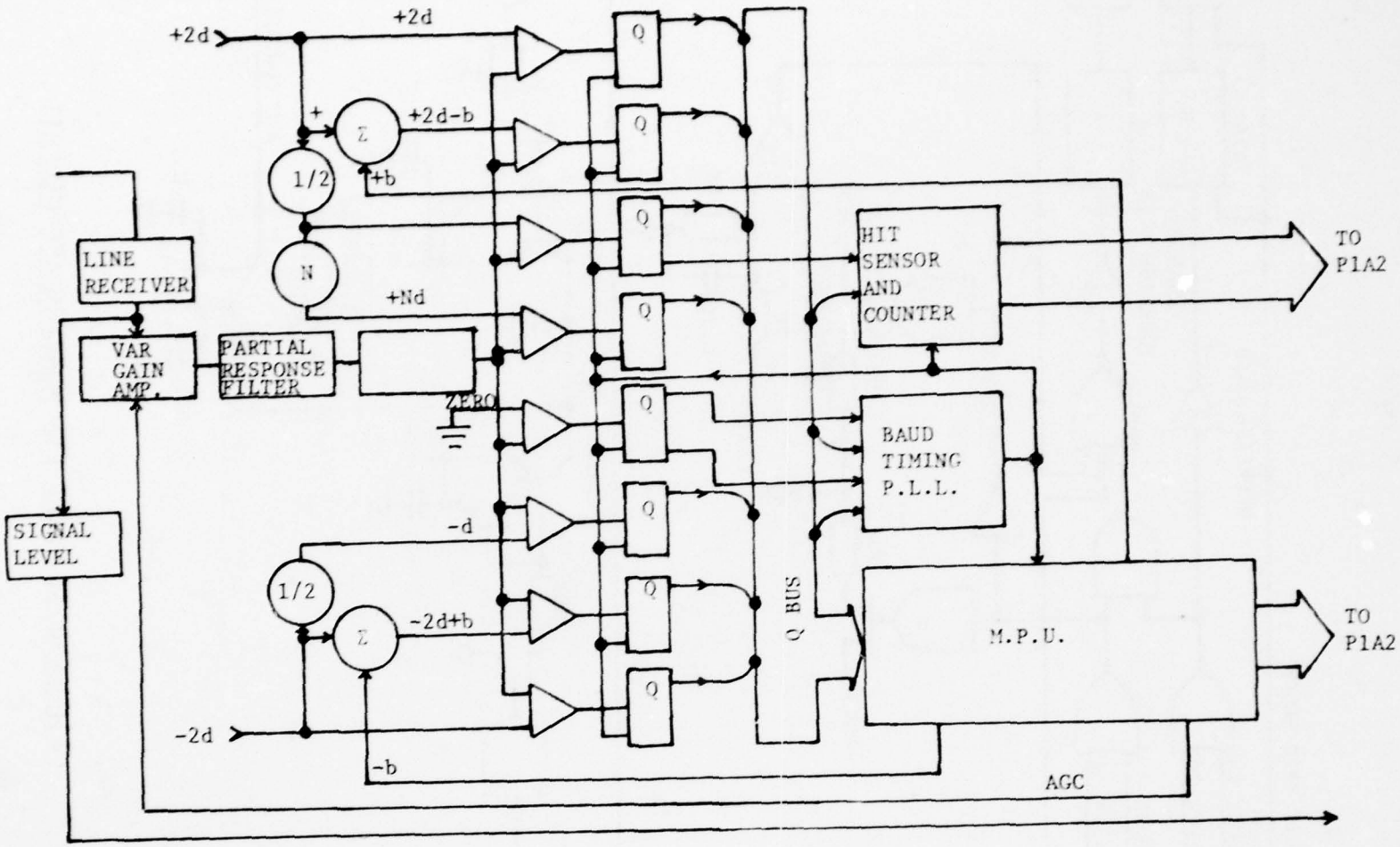


FIGURE 4.4: M-ATEC II EYE MONITOR

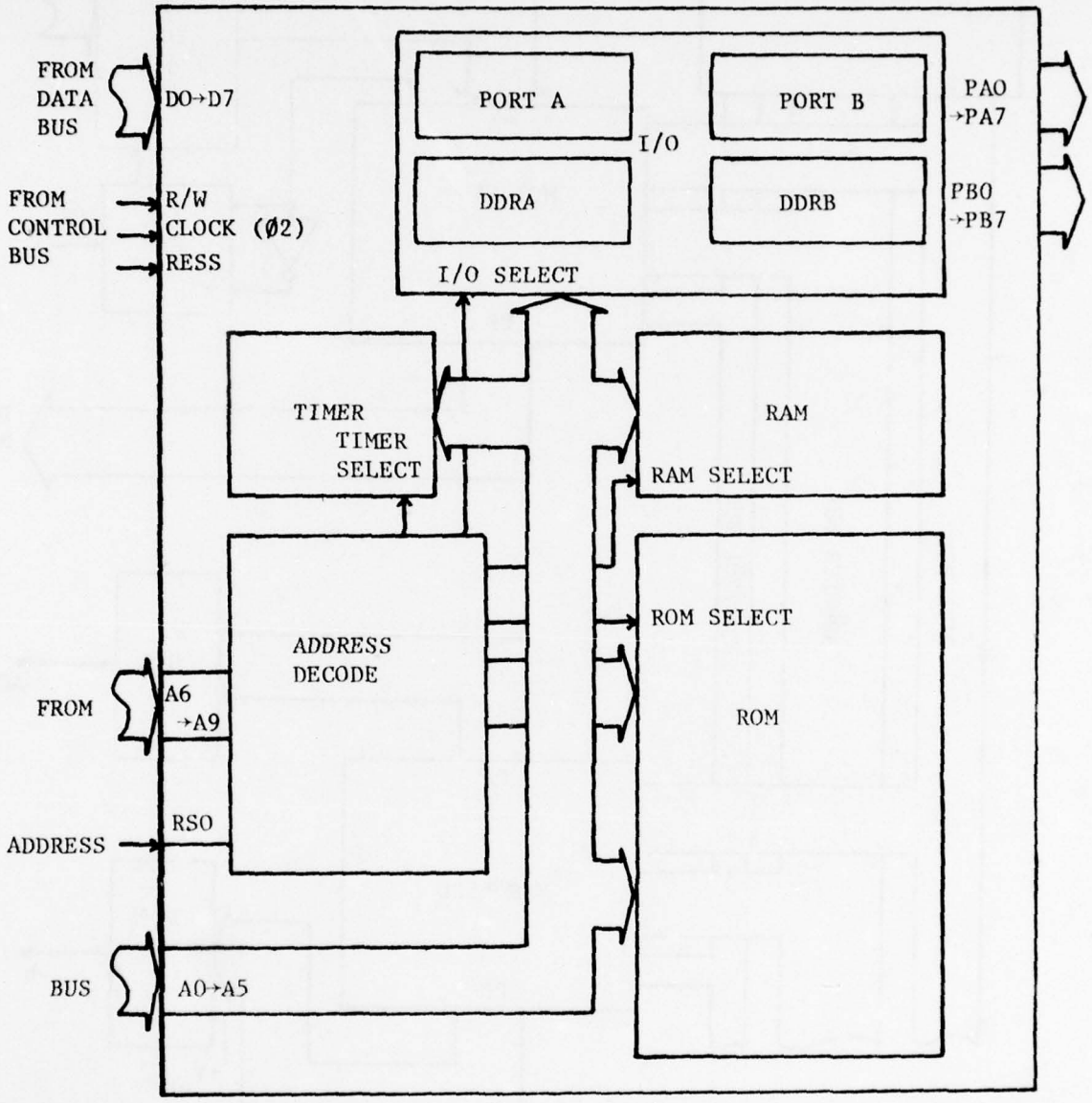


FIGURE 4.5: MCS 6530 INTERFACE/MEMORY DEVICE

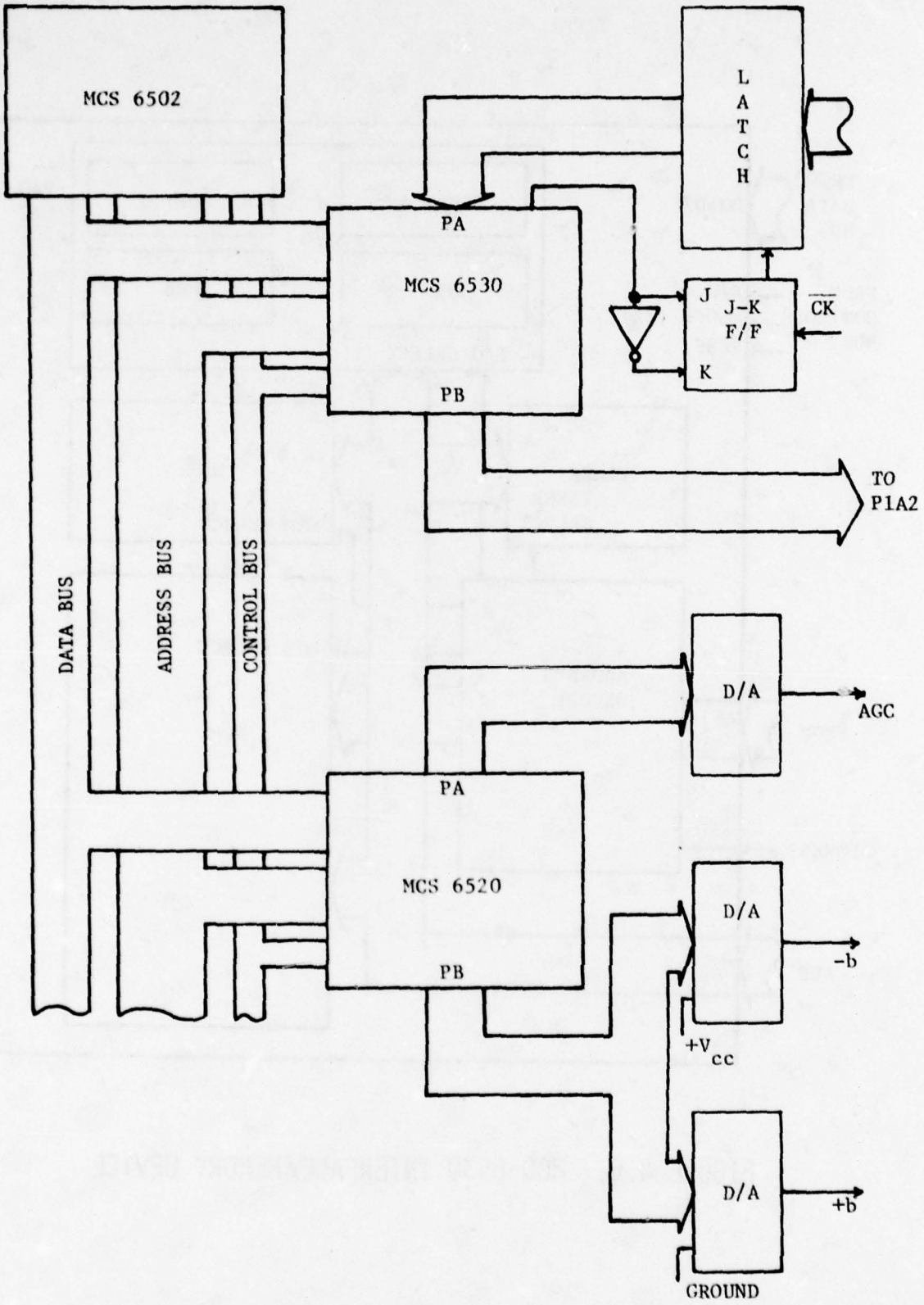


FIGURE 4.6: EYE MONITOR MPU HARDWARE

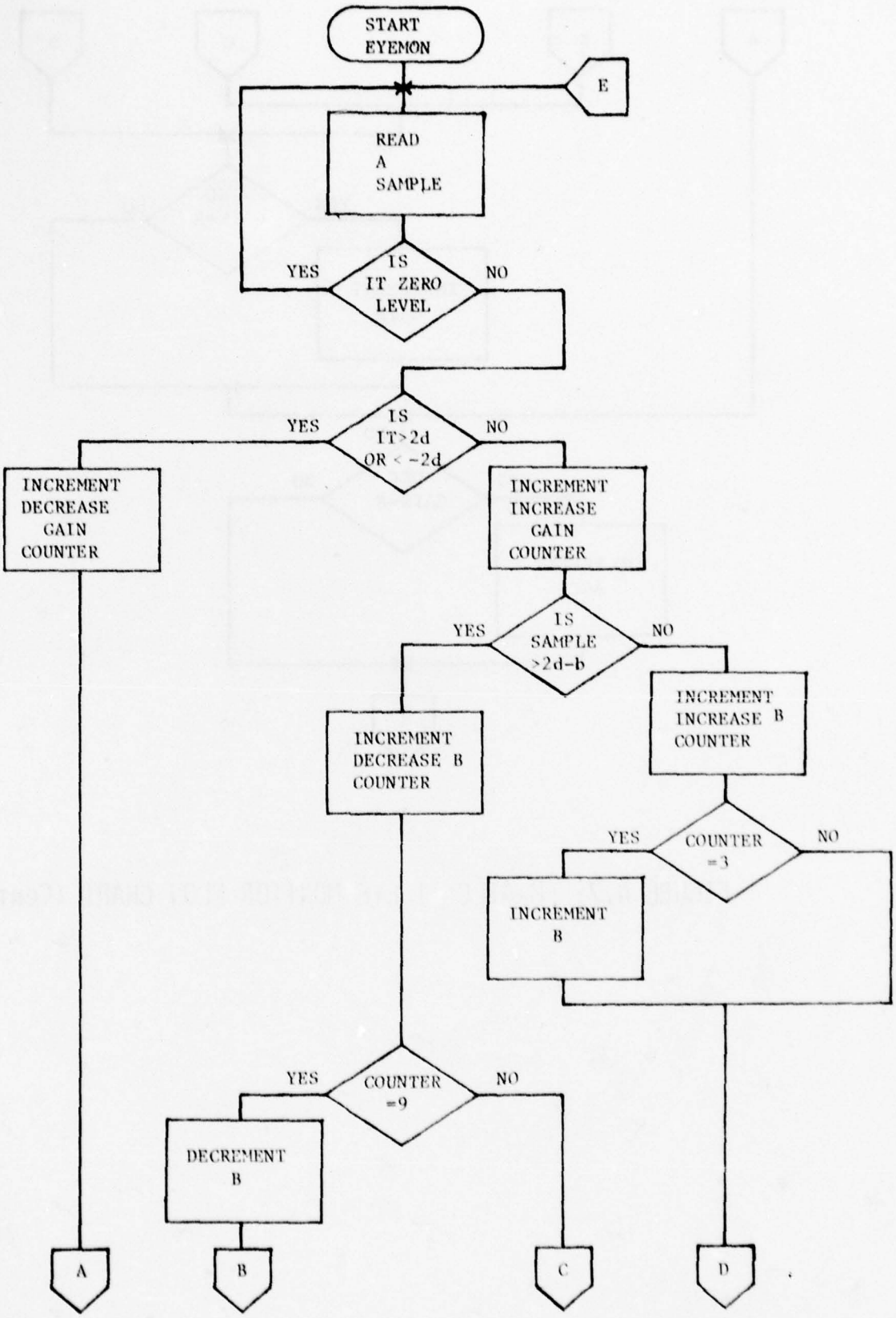


FIGURE 4.7: M-ATEC II EYE MONITOR FLOW CHART

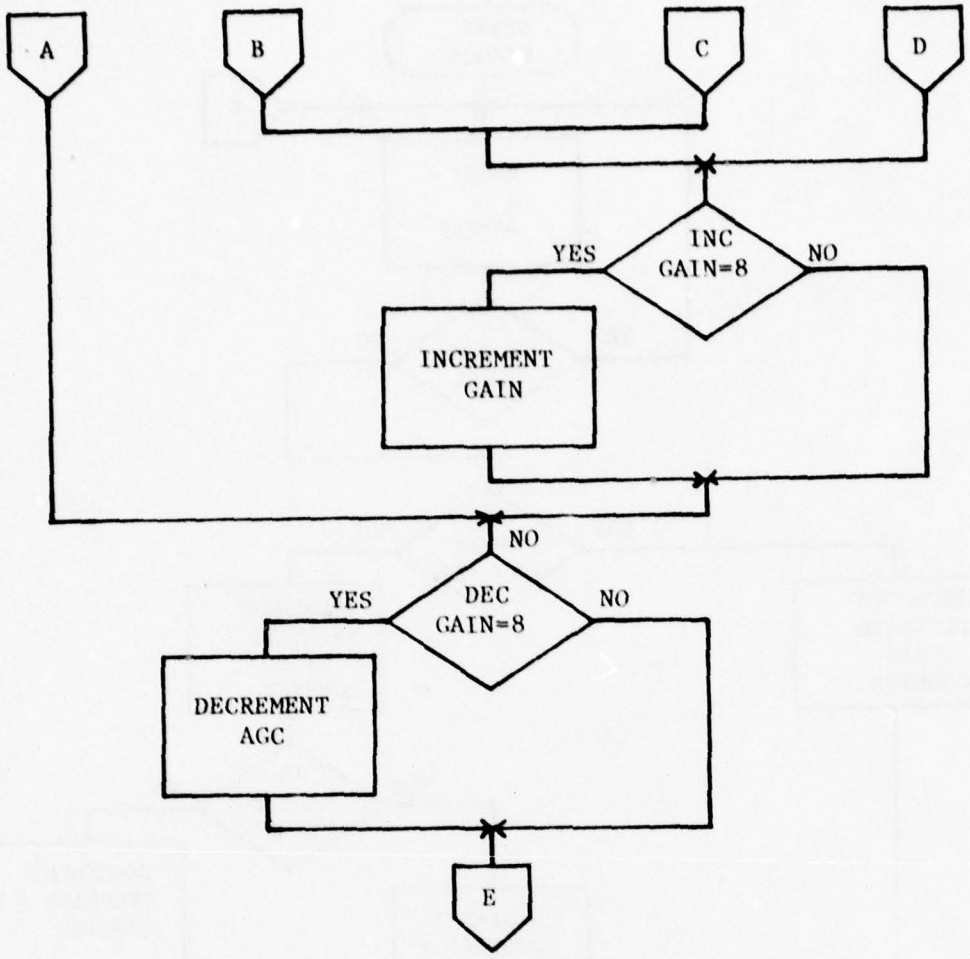


FIGURE 4.7: M-ATEC II EYE MONITOR FLOW CHART (CONT'D)

A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
****

SIMULATION SYSTEM CONSTANTS
****

THE RANDOM NUMBER GENERATOR SEED IS: 382
THE INITIAL VALUE OF THE AGC LEVEL IS: 1.0000
THE INITIAL VALUE OF THE B CONTROL LEVEL IS: 0.05000
THE RESOLUTION OF THE AGC IS: 0.10000
THE RESOLUTION OF THE B CONTROL IS: 0.05000
THE NUMBER OF SAMPLES SUGGESTING THAT THE AGC LEVEL SHOULD BE RAISED OR
LOWERED WHICH MUST BE COLLECTED BEFORE ANY CHANGE IS MADE IS: 8
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS DECREASED IS: 3
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS INCREASED IS: 9
THE DECISION THRESHOLD IS 1.00000

A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
****

SIMULATION SYSTEM RESULTS
****

IN THE GRAPH BELOW AGC, BCON & ACTUAL & MEASURED EYE OPENING ARE PLOTTED AGAINST TIME.
AGC IS REPRESENTED BY "A", EYE OPENING, AS MEASURED BY THE EYE PATTERN MONITOR,
BY "E" & INPUT SIGNAL LEVEL BY "Y".
THE CONTROL VOLTAGE B IS REPRESENTED BY "B".

THE POINT LEVEL SELECTOR HAS OUTER SAMPLES
A CURRENT SIGNAL SAMPLE VALUE OF 4.00000 HAS BEEN SELECTED.
A STEP CHANGE IS BEING APPLIED OF SIZE 0.50000 AT TIME 50000

FIGURE 4.8 (A): SYSTEM PARAMETERS FOR TEST 1

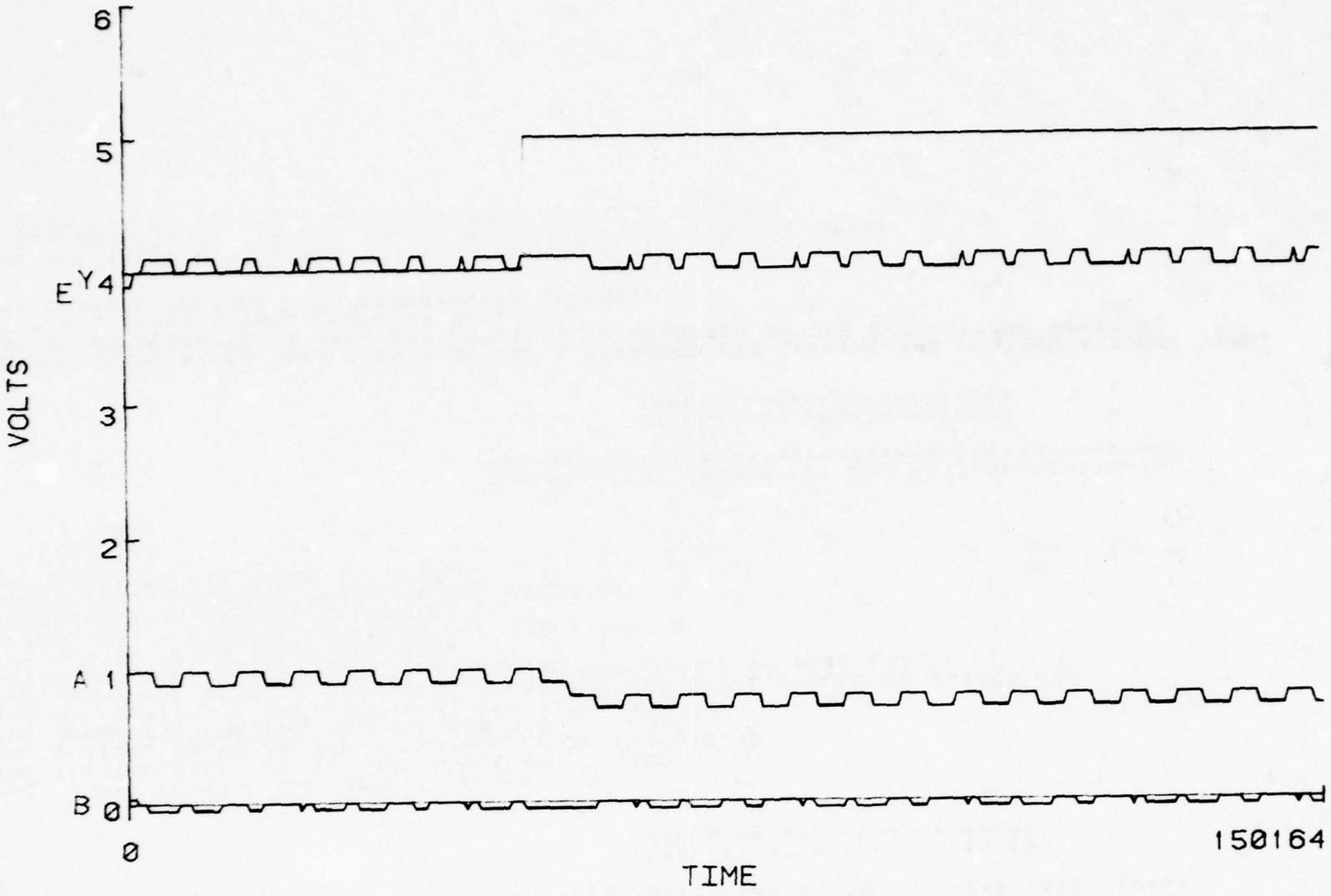


FIGURE 4.8 (B): RESULTS OF TEST 1

```
                    A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
                    **********************************************

                              SIMULATION SYSTEM CONSTANTS
                              ***************************


THE RANDOM NUMBER GENERATOR SEED IS:    505
THE INITIAL VALUE OF THE AGC LEVEL IS: 1.0000
THE INITIAL VALUE OF THE B CONTROL LEVEL IS: 0.05000
THE RESOLUTION OF THE AGC IS: 0.10000
THE RESOLUTION OF THE B CONTROL IS: 0.05000
THE NUMBER OF SAMPLES SUGGESTING THAT THE AGC LEVEL SHOULD BE RAISED OR
LOWERED WHICH MUST BE COLLECTED BEFORE ANY CHANGE IS MADE IS:   8
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS DECREASED IS:   3
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS INCREASED IS:   9
THE DECISION THRESHOLD IS 1.00000
```

```
                    A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
                    **********************************************

                              SIMULATION SYSTEM RESULTS
                              *************************

IN THE GRAPH BELOW AGC, RCCM & ACTUAL & MEASURED EYE OPENING ARE PLOTTED AGAINST TIME.
AGC IS REPRESENTED BY "A", EYE OPENING, AS MEASURED BY THE EYE PATTERN MONITOR,
BY "E" & INPUT SIGNAL LEVEL BY "Y".
THE CONTROL VOLTAGE B IS REPRESENTED BY "B".

THE PRINT LEVEL SELECTED WAS OUTER SAMPLES
A CONSTANT SIGNAL SAMPLE VALUE OF 2.00000 HAS BEEN SELECTED.
A STEP CHANGE IS BEING ADDED OF SIZE -0.80000 AT TIME       50000
```


FIGURE 4.9 (A): SYSTEM PARAMETERS FOR TEST 2

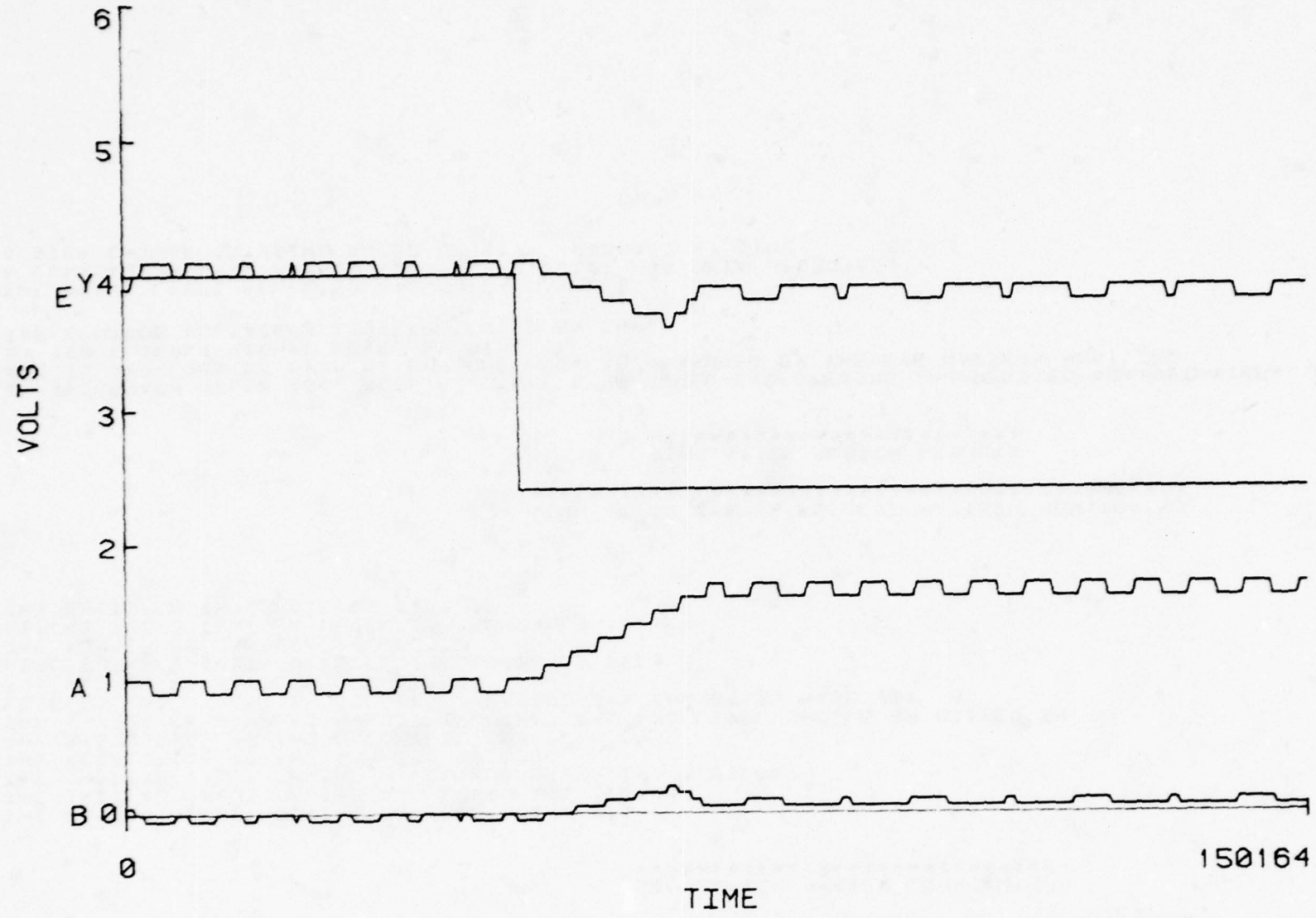


FIGURE 4.9 (B): RESULTS OF TEST 2

A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
****

SIMULATION SYSTEM CONSTANTS
****

THE RANDOM NUMBER GENERATOR SEED IS: 256
THE INITIAL VALUE OF THE AGC LEVEL IS: 1.0000
THE INITIAL VALUE OF THE B CONTROL LEVEL IS: 0.50000
THE RESOLUTION OF THE AGC IS: .1000
THE RESOLUTION OF THE B CONTROL IS: .05000
THE NUMBER OF SAMPLES SUGGESTING THAT THE AGC LEVEL SHOULD BE RAISED OR
LOWERED WHICH MUST BE COLLECTED BEFORE ANY CHANGE IS MADE IS: 8
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS DECREASED IS: 3
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS INCREASED IS: 9
THE DECISION THRESHOLD IS 1.0000

A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
****

SIMULATION SYSTEM RESULTS
****

IN THE GRAPH BELOW AGC, ROOM & ACTUAL & MEASURED EYE OPENING ARE PLOTTED AGAINST TIME.
AGC IS REPRESENTED BY "A", EYE OPENING, AS MEASURED BY THE EYE PATTERN MONITOR,
BY "E" & INPUT SIGNAL LEVEL BY "Y".
THE CONTROL VOLTAGE B IS REPRESENTED BY "B".

THE POINT LEVEL SELECTED WAS OUTER SAMPLES
A CONSTANT SIGNAL SAMPLE VALUE OF 2.00000 HAS BEEN SELECTED.
A SAMPLE CHANGE OF SLOPE -0.00125 STARTING AT 50000 AND FINISHING AT 80000
IS BEING ADDED.

FIGURE 4.10 (A): SYSTEM PARAMETER FOR TEST 3

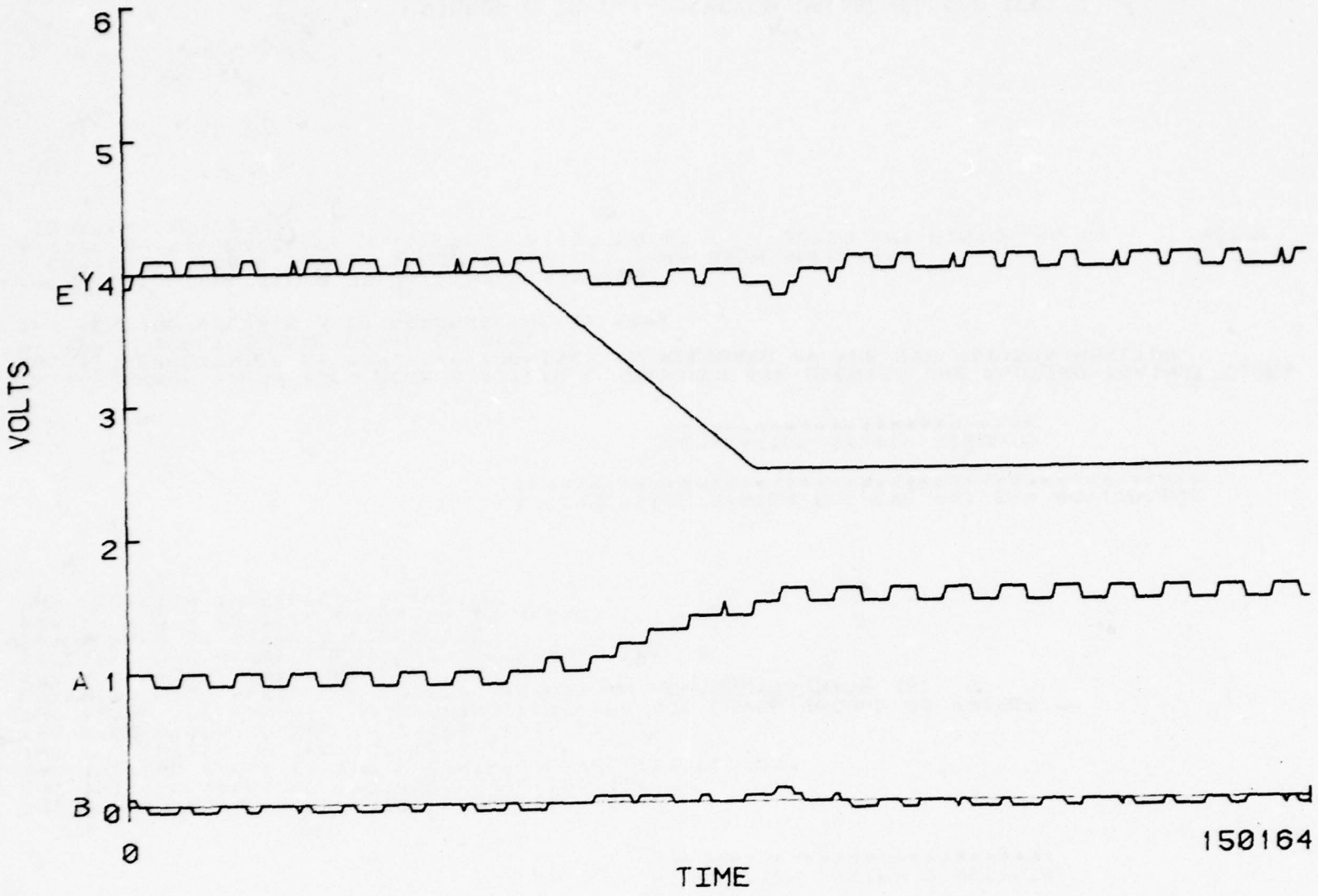


FIGURE 4.10 (B): RESULTS OF TEST 3

A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
****

SIMULATION SYSTEM CONSTANTS
****

THE RANDOM NUMBER GENERATOR SEED IS: 270
THE INITIAL VALUE OF THE AGC LEVEL IS: 1.0000
THE INITIAL VALUE OF THE B CONTROL LEVEL IS: 0.05000
THE RESOLUTION OF THE AGC IS: 0.10000
THE RESOLUTION OF THE B CONTROL IS: 0.05000
THE NUMBER OF SAMPLES SUGGESTING THAT THE AGC LEVEL SHOULD BE RAISED OR
LOWERED WHICH MUST BE COLLECTED BEFORE ANY CHANGE IS MADE IS: 8
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS DECREASED IS: 3
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS INCREASED IS: 9
THE DECISION THRESHOLD IS 1.00000

A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
****

SIMULATION SYSTEM RESULTS
****

IN THE GRAPH BELOW AGC, BOOM & ACTUAL & MEASURED EYE OPENING ARE PLOTTED AGAINST TIME.
AGC IS REPRESENTED BY "A", EYE OPENING, AS MEASURED BY THE EYE PATTERN MONITOR,
BY "E" & INPUT SIGNAL LEVEL BY "Y".
THE CONTROL VOLTAGE B IS REPRESENTED BY "B".

THE PRINT LEVEL SELECTED WAS OUTER SAMPLES
A CONSTANT SIGNAL SAMPLE VALUE OF 2.00000 HAS BEEN SELECTED.

GAUSSIAN NOISE IS BEING ADDED TO THE SAMPLES, THE VARIANCE OF THE NOISE IS 0.25000
A RAMP CHANGE OF SLOPE -0.0000250 STARTING AT 50000 AND FINISHING AT 80000
IS BEING ADDED.

FIGURE 4.11 (A): SYSTEM PARAMETERS FOR TEST 4

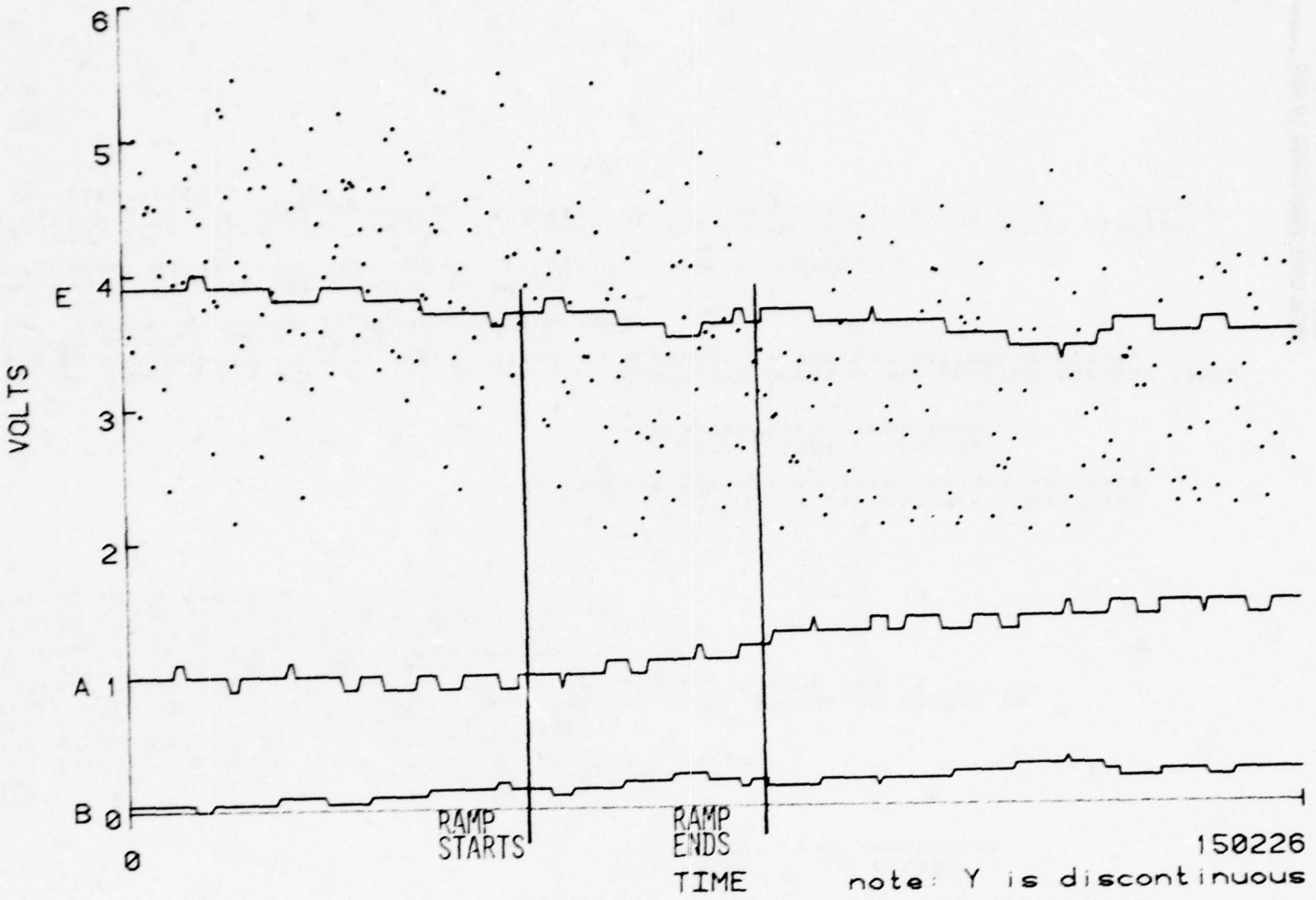


FIGURE 4.11 (B): RESULTS OF TEST 4

```
                    A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
                    **********************************************

                                SIMULATION SYSTEM CONSTANTS
                                ***************************

THE RANDOM NUMBER GENERATOR SEED IS:    510
THE INITIAL VALUE OF THE AGC LEVEL IS: 1.0000
THE INITIAL VALUE OF THE B CONTROL LEVEL IS: 0.05000
THE RESOLUTION OF THE AGC IS: 0.10000
THE RESOLUTION OF THE B CONTROL IS: 0.05000
THE NUMBER OF SAMPLES SUGGESTING THAT THE AGC LEVEL SHOULD BE RAISED OR
LOWERED WHICH MUST BE COLLECTED BEFORE ANY CHANGE IS MADE IS:  8
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS DECREASED IS:  3
THE NUMBER OF SAMPLES COLLECTED
BEFORE THE B CONTROL LEVEL IS INCREASED IS:  9
THE DECISION THRESHOLD IS 1.00000
```

```
                    A SIMULATION SYSTEM FOR EYE PATTERN MONITORING
                    **********************************************

                                SIMULATION SYSTEM RESULTS
                                *************************

IN THE GRAPH BELOW AGC, BCON & ACTUAL & MEASURED EYE OPENING ARE PLOTTED AGAINST TIME.
AGC IS REPRESENTED BY "A", EYE OPENING, AS MEASURED BY THE EYE PATTERN MONITOR,
BY "E" & INPUT SIGNAL LEVEL BY "Y".
THE CONTROL VOLTAGE B IS REPRESENTED BY "B".

THE PRINT LEVEL SELECTED WAS OUTER SAMPLES
A CONSTANT SIGNAL SAMPLE VALUE OF 2.00000 HAS BEEN SELECTED.
PERIODIC FADING OF FREQUENCY  0.00010 IS BEING ADDED
```


FIGURE 4.12 (A): SYSTEM PARAMETERS FOR TEST 5

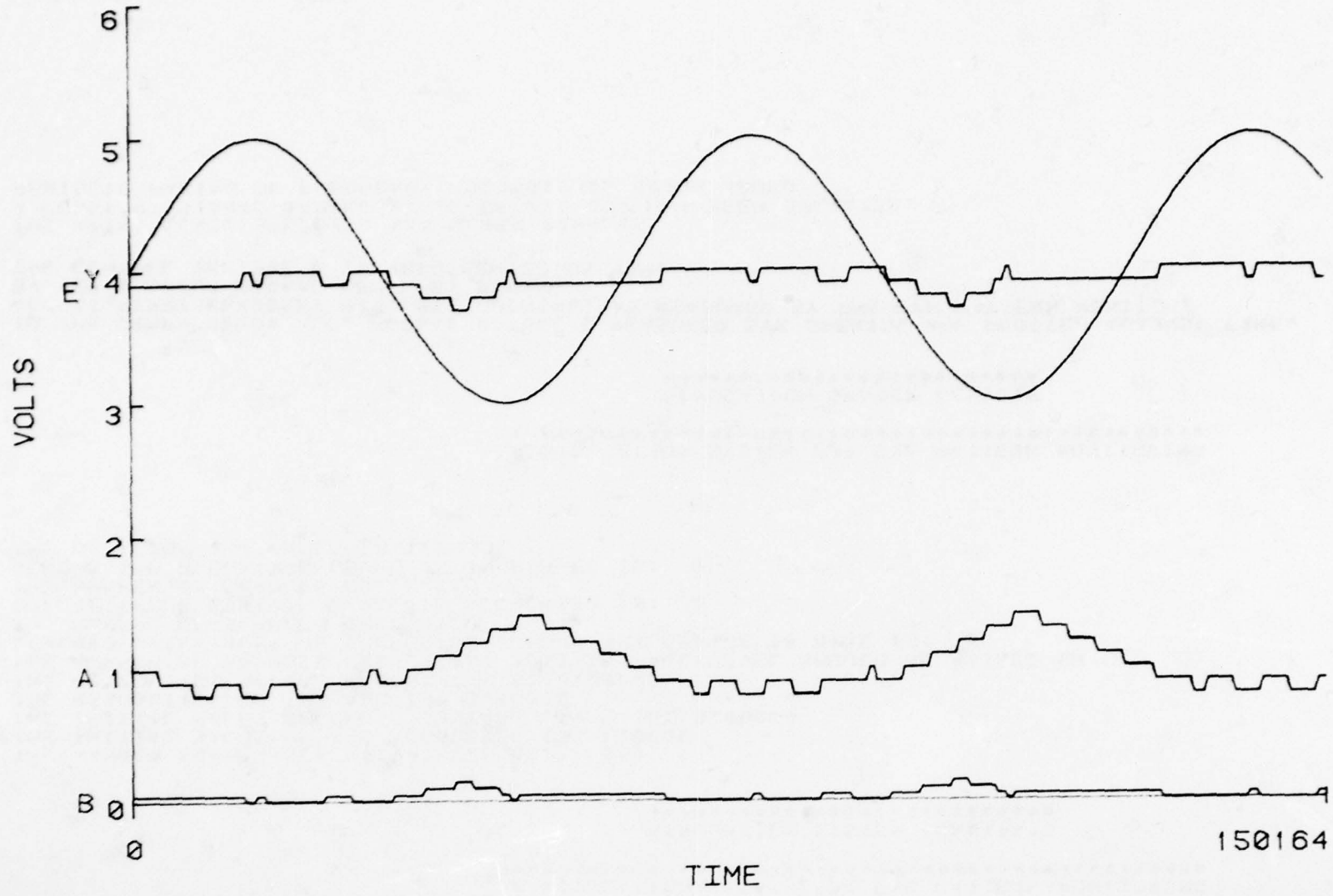


FIGURE 4.12 (B): RESULTS OF TEST 5

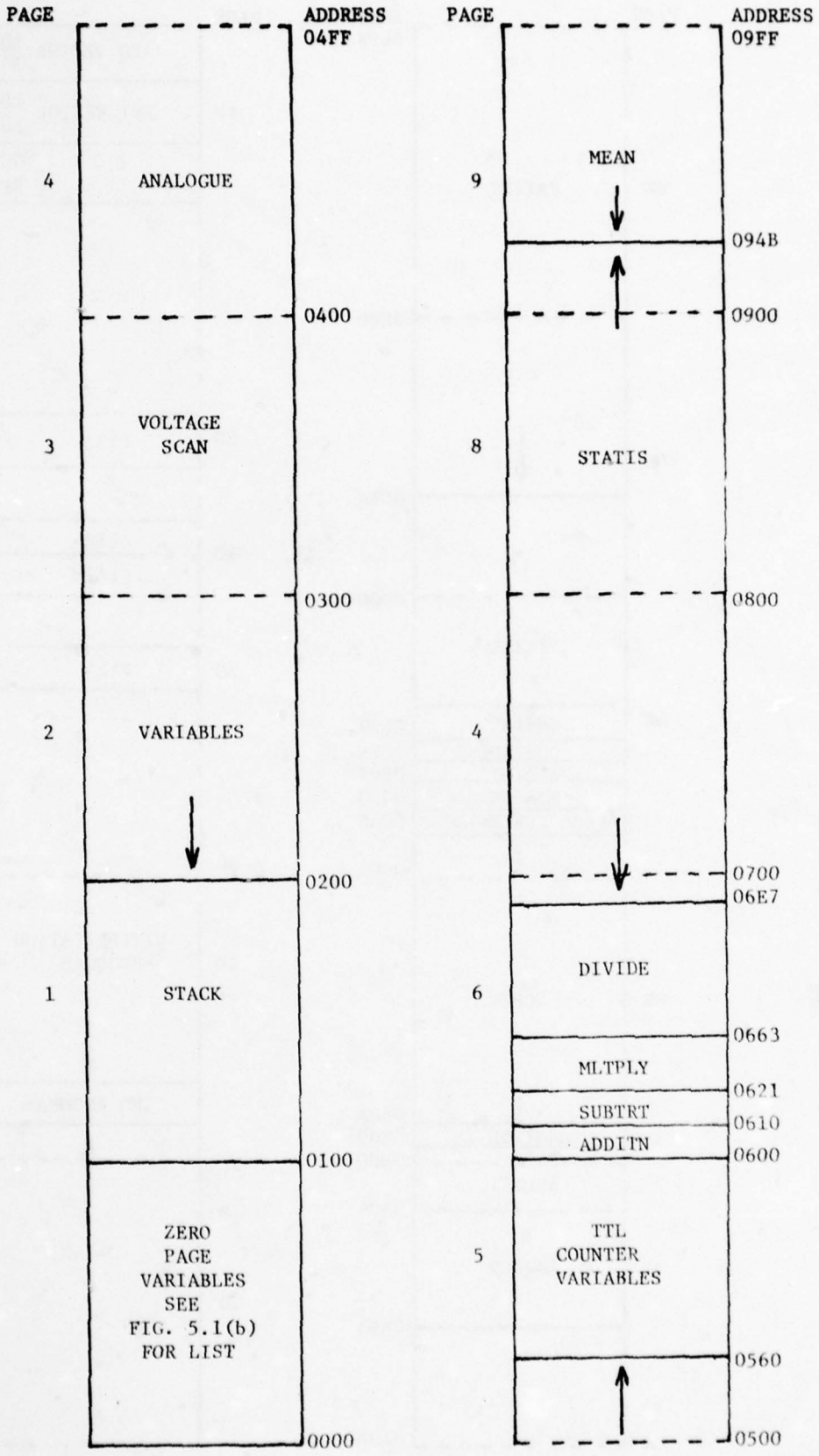


FIGURE 5.1(A): M-ATEC II MEMORY MAP

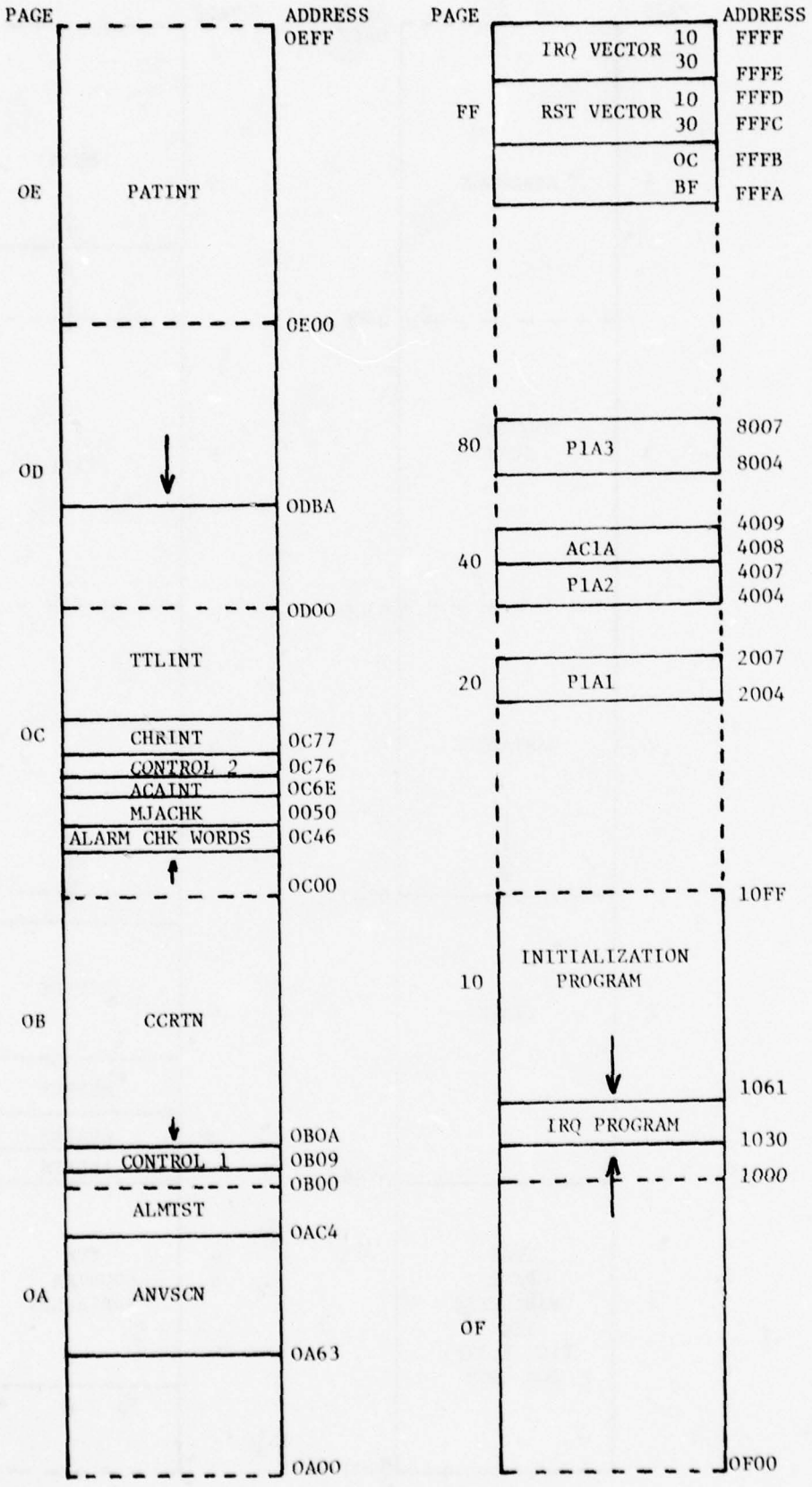


FIGURE 5.1(A): M-ATEC II MEMORY MAP (CONT'D)

ADDRESS	VARIABLE
1D	NACKFLAG
1C	CONCNT
1B	CNTN
1A	TSBN
19	ANCTR
18	BLKCNT
17	SAMCNT
16	POINTH3
15	POINTL3
14	POINTH2
13	POINTL2
12	POINTH1
11	POINTL1
10	SIZE
OF	MSIZE
OE	MULP+2
OD	MULP+1
OC	MULP
OB	AUG+3
OA	AUG+2
09	AUG+1
08	AUG
07	ADD+3
06	ADD+2
05	ADD+1
04	ADD
03	TEMP4
02	TEMP3
01	TEMP2
00	TEMP1

ADDRESS	VARIABLE
28	TXPNT
27	AFLAG
26	CCLCN8
25	CCLCN7
24	CCLCN6
23	CCLCN5
22	CCLCN4
21	CCLCN3
20	CCLCN2
1F	CCLCN1
1E	CCLCN0

FIGURE 5.1(B): ZERO PAGE VARIABLES

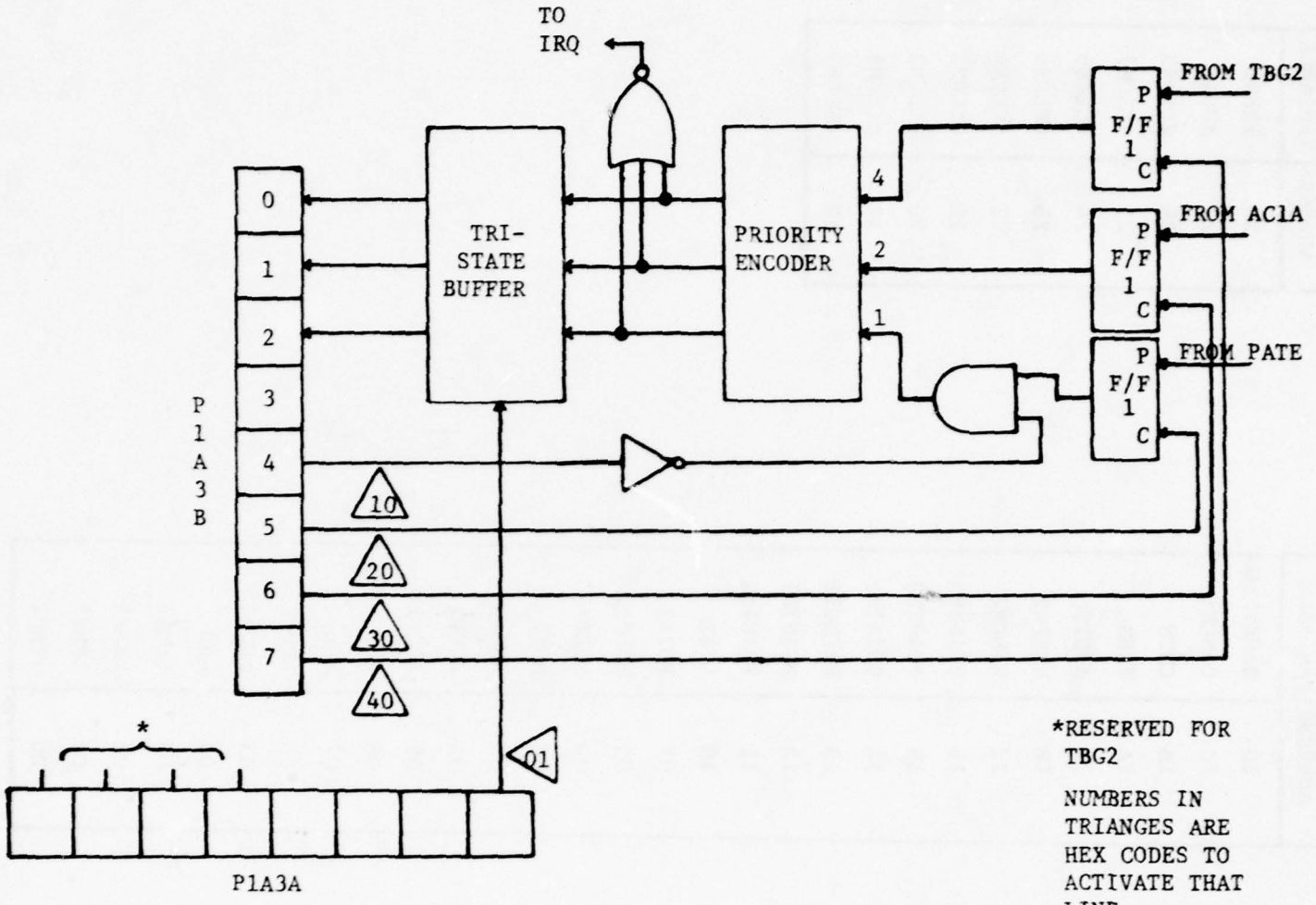


FIGURE 5.2: M-ATEC II INTERRUPT PRIORITY CONTROL

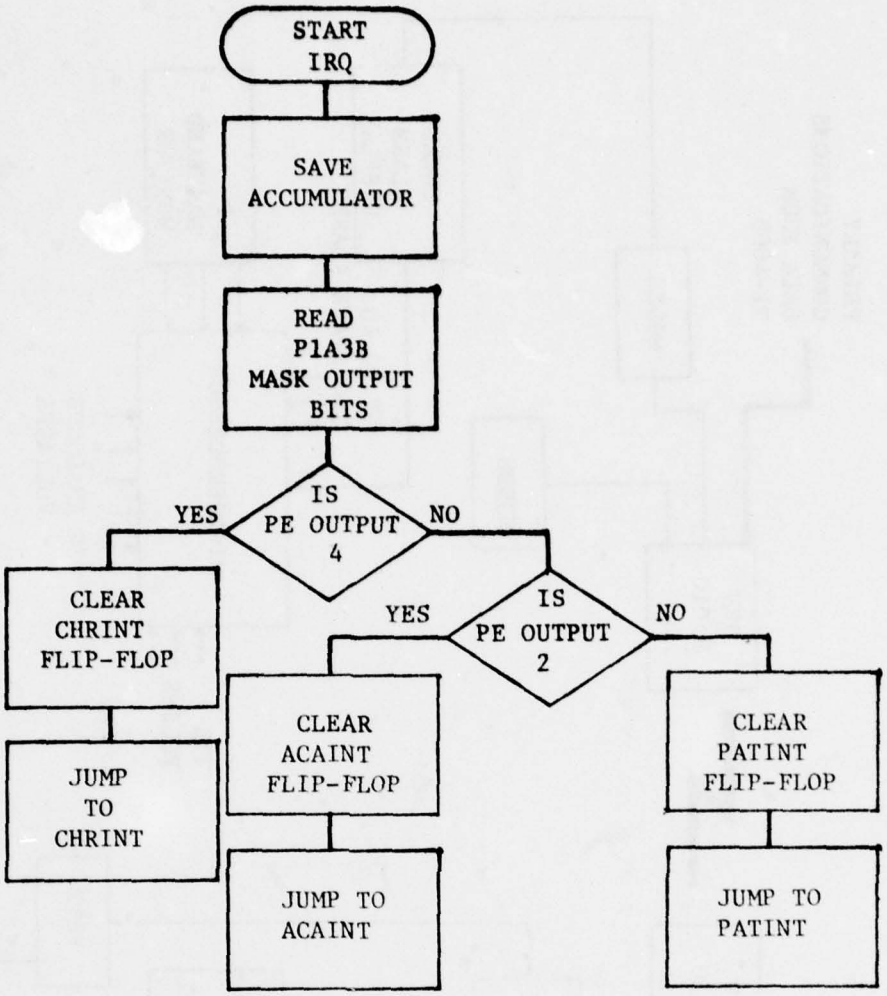


FIGURE 5.3: IRQ SERVICE PROGRAM FLOW CHART

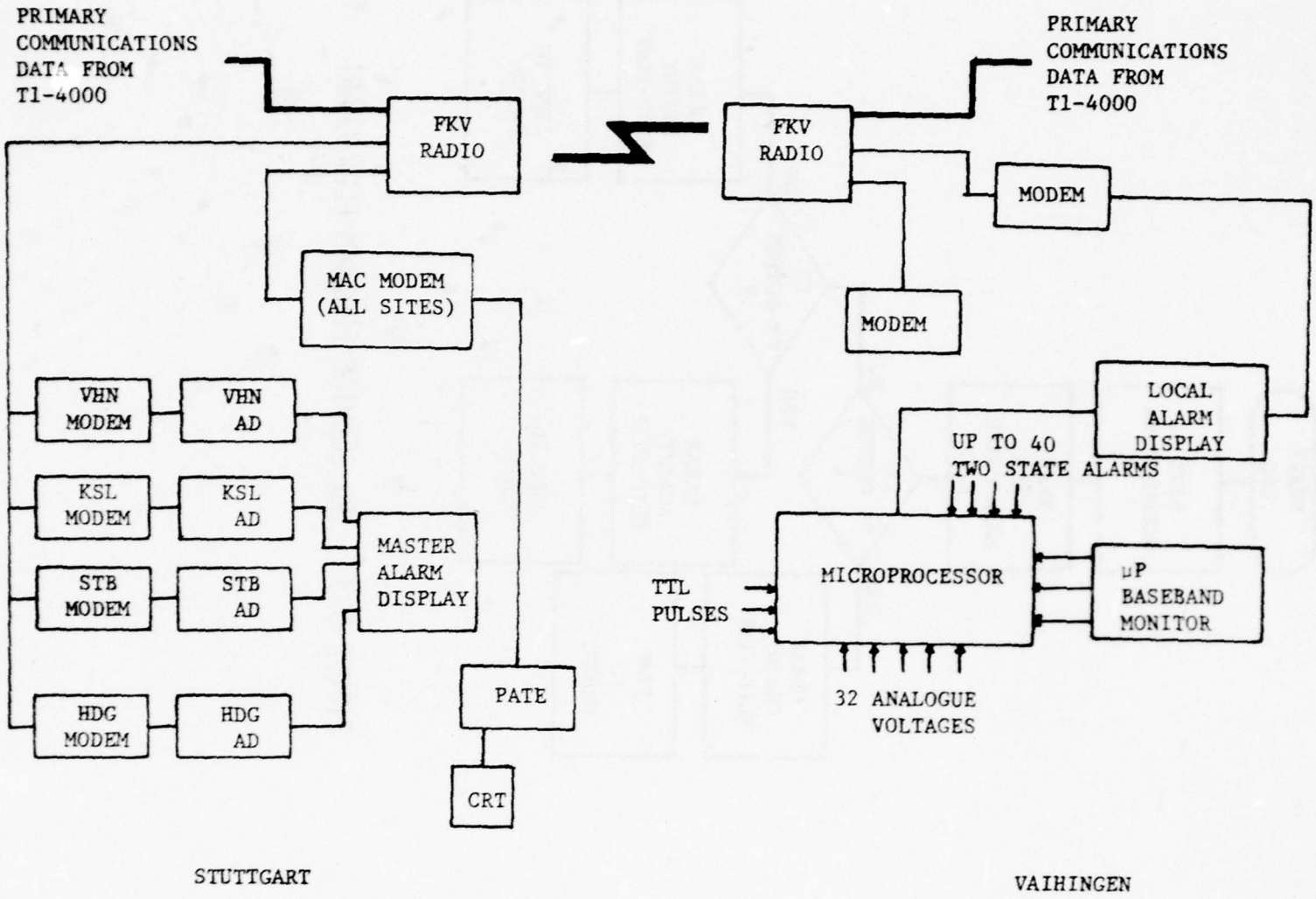


FIGURE 5.4: M-ATEC II HARDWARE CONFIGURATION

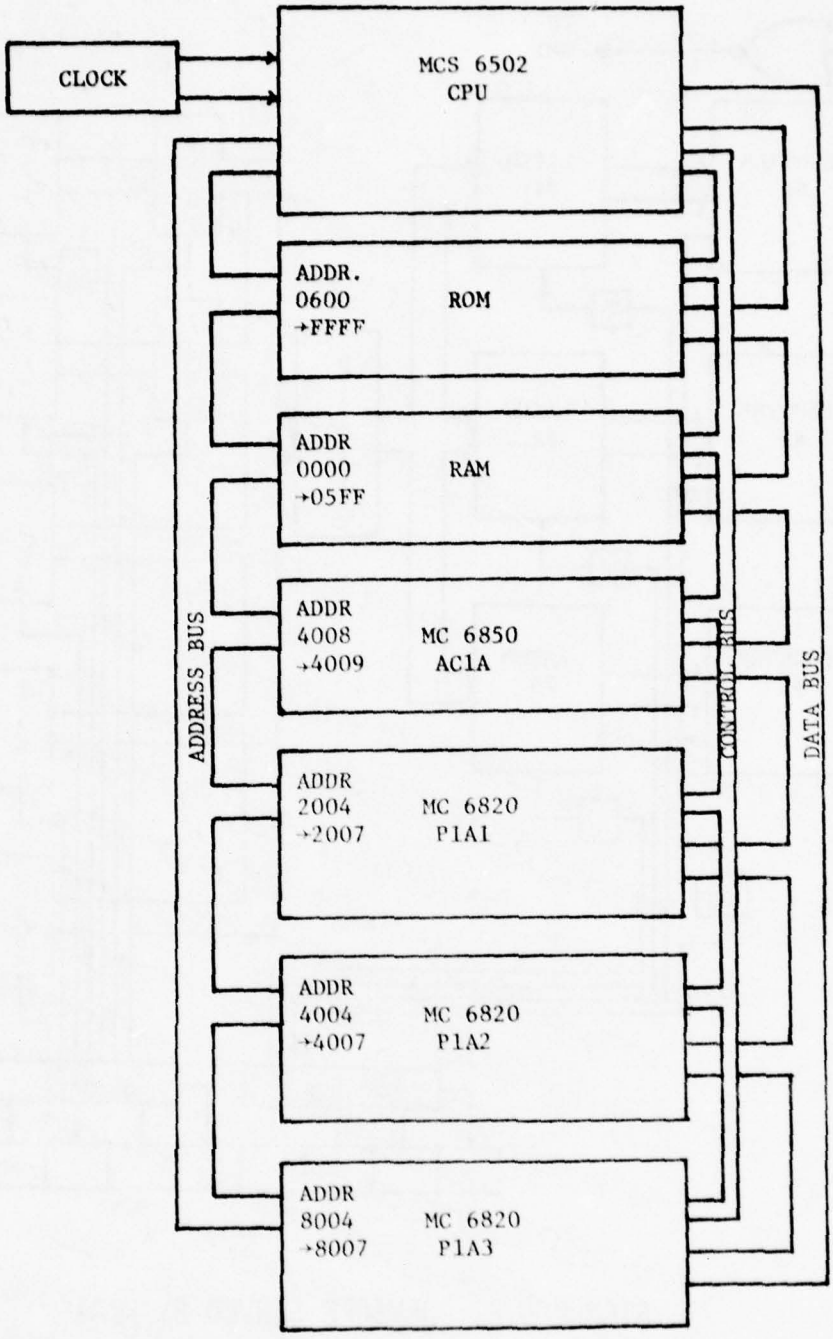


FIGURE 5.5: M-ATEC II MICROCOMPUTER HARDWARE

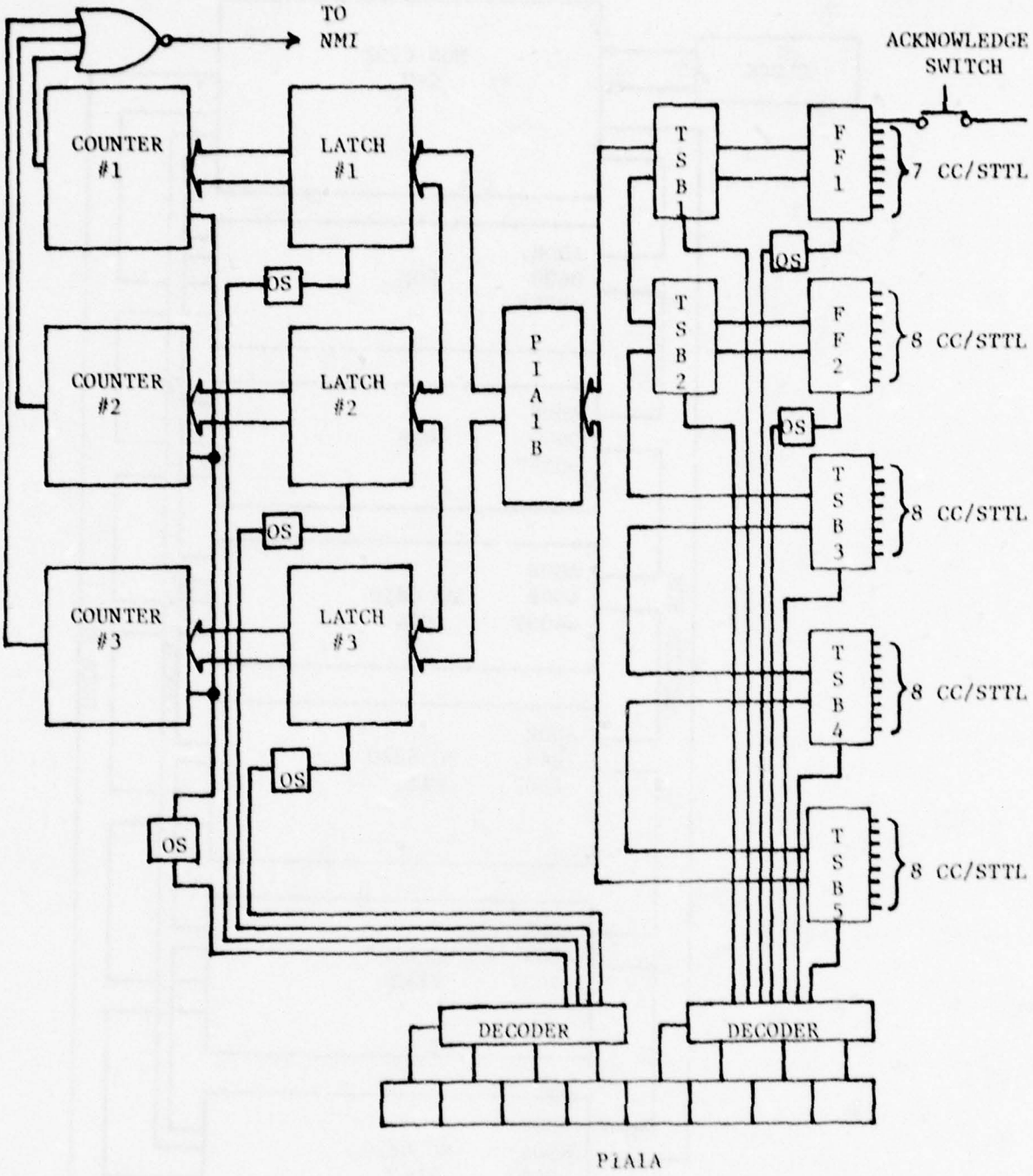


FIGURE 5.6: DEVICES SERVED BY PIA1
(STTL/CC, ACKNOWLEDGE SWITCH, TBG1)

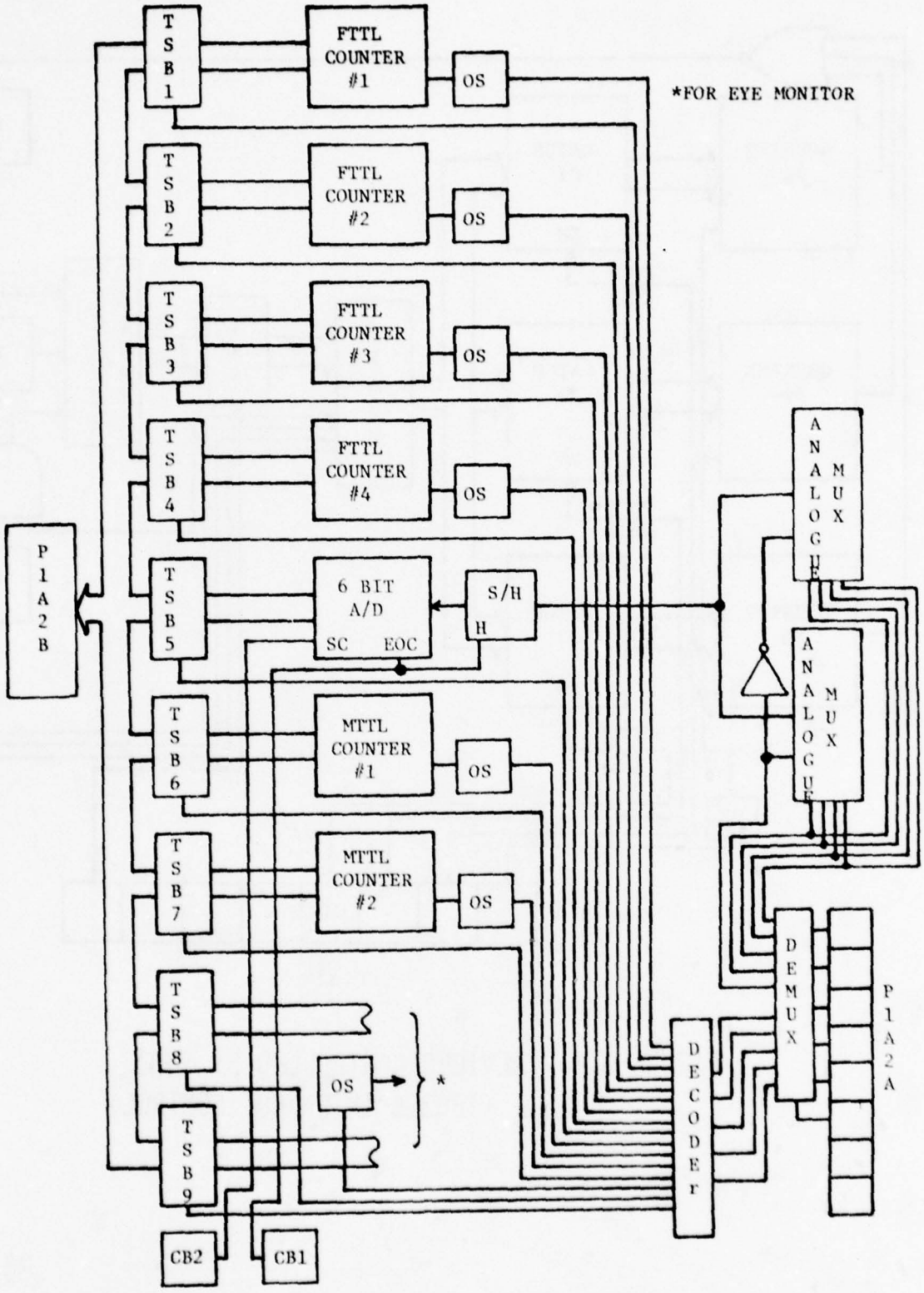


FIGURE 5.7: DEVICES SERVED BY PIA2 (TTL COUNTERS, ANVSCN, EYE MONITOR)

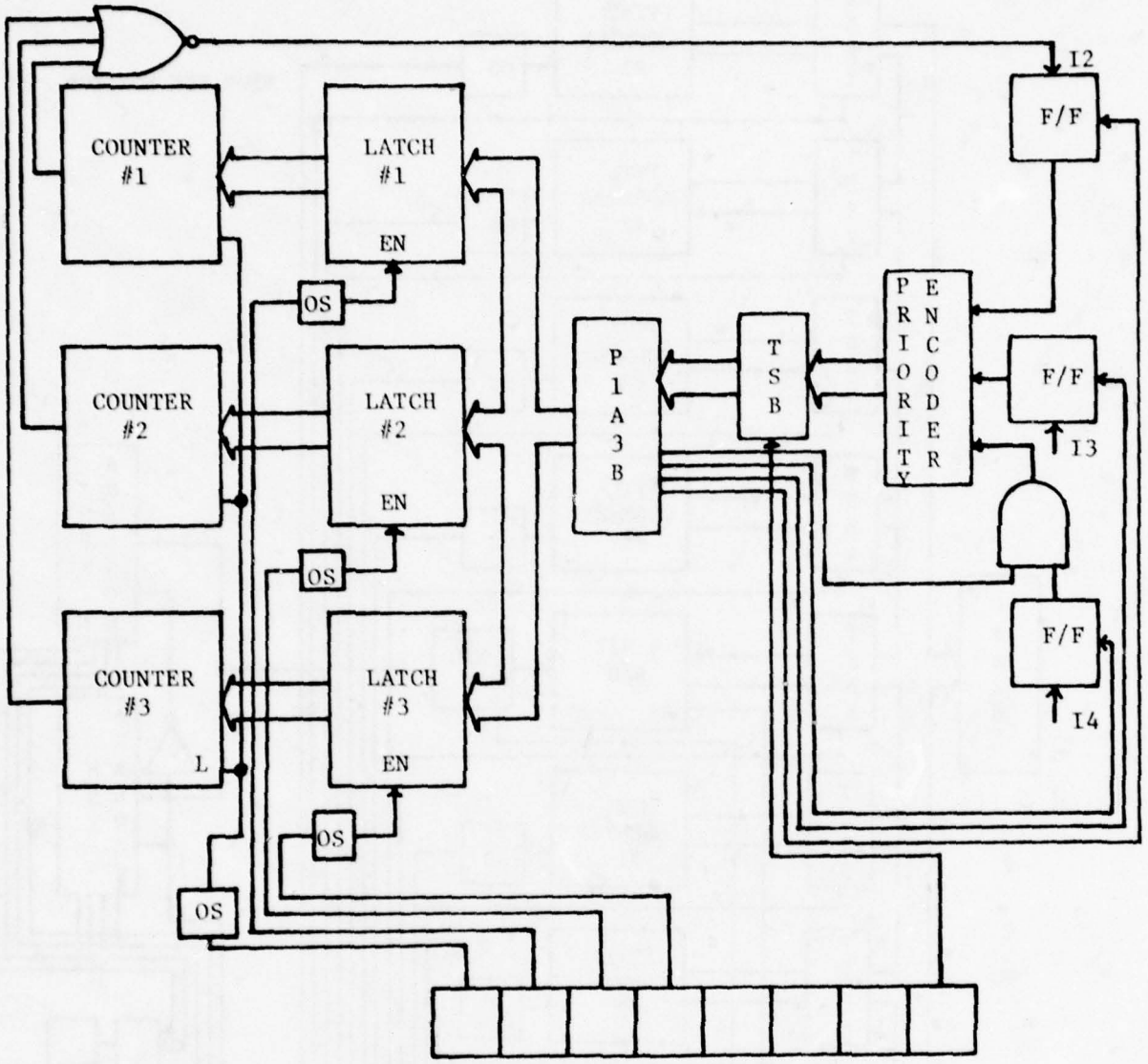


FIGURE 5.8: DEVICES CONTROLLED BY PIA3
(TBG2 & INTERRUPT CONTROL)

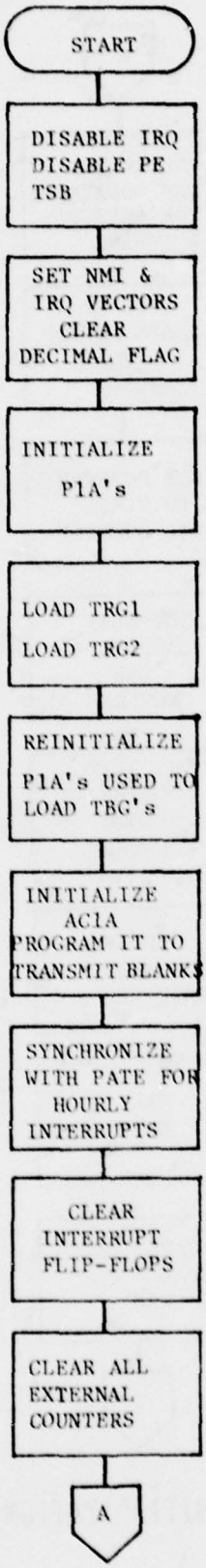


FIGURE 5.9: INITIALIZATION FLOW CHART

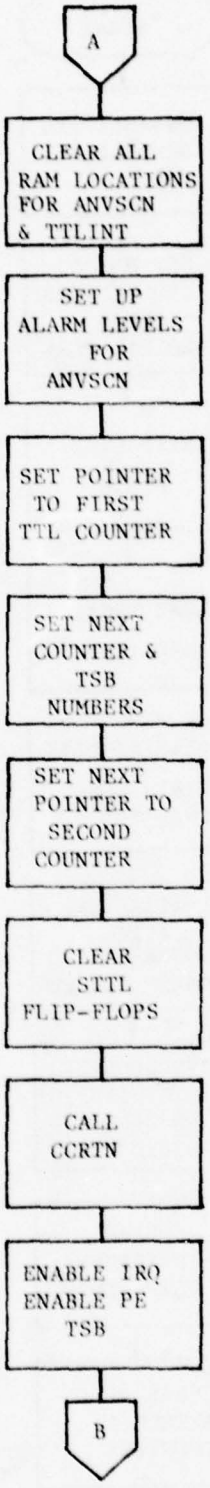


FIGURE 5.9: SYSTEM INITIALIZATION FLOW CHART (CONT'D)

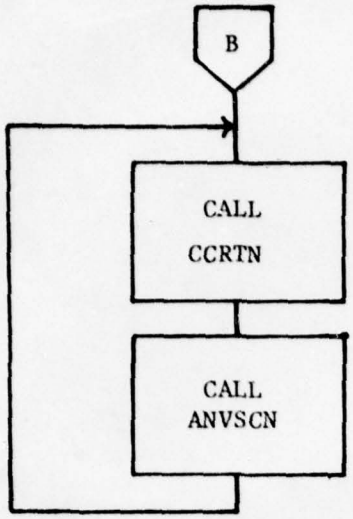


FIGURE 5.9: SYSTEM INITIALIZATION FLOW CHART (CONT'D)

APPENDIX A

DERIVATION OF STATISTICS EQUATIONS

By definition the mean, $m = \frac{\sum_{i=1}^{n} x_i}{n}$ -(1) here x_i are the samples

n is the number of samples

and the variance $\sigma^2 = \frac{\sum_{i=1}^{n} (x_i - m)^2}{n}$

$$\therefore \quad \sigma^2 = \frac{\sum_{i=1}^{n} (x_i - m)\,(x_i - m)}{n} \qquad -(2)$$

$$\therefore \quad n\sigma^2 = \sum_{i=1}^{n} (x_i)^2 + \sum_{i=1}^{n} (m)^2 - \sum_{i=1}^{n} 2mx_i$$

$$= \sum_{i=1}^{n} (x_i)^2 + \frac{n\sum_{i=1}^{n} (x_i)^2}{n^2} - \frac{2\sum_{i=1}^{n} x_i^2}{n}$$

$$\therefore \quad \sigma^2 = \frac{\sum_{i=1}^{n} (x_i)^2}{n} - \frac{(\sum_{i=1}^{n} x_i)^2}{n^2} \qquad -(3)$$

Equation (3) is a useful expression for calculating the variance. The overall mean of several equal size blocks of samples, m_{or}, is given by

$$m_{ov} = \frac{\sum_{i=1}^{n} x_i}{N}$$ where N = q x n is the total number of samples q is the number of blocks of samples

$$= \frac{\sum_{i=1}^{n} x_i + \sum_{i=n+1}^{2n} x_i + \ldots + \sum_{i=qn-n+1}^{qn} x_i}{\sum_{j=1}^{q} n}$$

$$= \frac{nm_1 + nm_2 + \ldots . nm_q}{qn} \qquad -(4)$$

$$\therefore \quad m_{ov} = \frac{\sum_{i=1}^{q} m_i}{q} \qquad -(5)$$

To find a similar formula for calculating the variance it is first necessary to find an expression for the overall mean in terms of the mean of the latest block of samples plus a sum of the other means.

From equation (4)

$$m_{ov} = \frac{n_1 m_1}{N} + \frac{n_2 m_2}{N} + \ldots\ldots + \frac{n_q m_q}{N}$$

In the above equation

$$n_j = N - \sum_{\substack{i=1 \\ i \neq j}}^{q} n_i$$

$$\therefore \quad m_{ov} = \frac{(N - \sum_{\substack{i=1 \\ i \neq j}}^{q} n_i)\, m_j}{N} + \frac{\sum_{\substack{i=1 \\ i \neq q}}^{q} n_i m_i}{N}$$

$$= \frac{m_j (1 - \sum_{i=1}^{q} n_i)}{N} + \frac{1}{N} \sum_{\substack{i=1 \\ i \neq j}}^{q} n_i m_i$$

$$= m_j + \frac{\sum_{i=1, i \neq j}^{q} n_i (m_i - m_j)}{N}$$

but if $i=j$ then $m_i - m_j = 0$

$$\therefore \quad m_{ov} = m_j + \frac{\sum_{i=1}^{q} n_i (m_i - m_j)}{N} \qquad -(6)$$

Now define $\Sigma_{(K)} = \sum_{i=\sum_{1}^{k-1} n_k + 1}^{\sum_{1}^{k} n_k}$

Since $\sigma^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - m)^2$

$$\therefore \quad n\sigma^2 = \sum_{i=1}^{n} (x_i - m)^2$$

then the overall variance for N samples where $N = n_1 + n_2 + \ldots\ldots + n_q$ is given by:

$$N\sigma^2 = \sum_{K=1}^{q} [\Sigma_{(K)} (x_i - M_{ov})] \text{ where } M_{or} = \text{overall mean of the N samples.}$$

If K=j then substituting equation (6) for M_{ov} gives

$$\Sigma_{i(j)} (x_i - M_{or}) = \Sigma_{i(j)} [x_i - m_j - \frac{1}{N} \sum_{\ell=1}^{q} n_\ell (m_\ell - m_j)]^2$$

$$= \Sigma_{i(j)} (x_i - m_j)^2 - \frac{2}{N} \sum_{\ell=1}^{q} n_\ell (m_\ell - m_j)$$

$$+ \frac{1}{N^2} [\sum_{\ell=1}^{q} n_\ell (m_\ell - m_j)]$$

Since the mean is the balance point of the distribution $\Sigma (x_i - m_j)$ is zero.

$$\therefore \quad \Sigma_{i(j)} (x_i - m)^2 = \Sigma_{i(j)} (x_i - m_j)^2 + \Sigma_{i(j)} \frac{1}{N^2} [\sum_{\ell=1}^{q} n_\ell (m_\ell - m_j)]^2$$

$$= \Sigma_{i(j)} (x_i - m_j)^2 + \frac{n_j}{N^2} [\sum_{\ell=1}^{q} n_\ell (m_\ell - m_j)]^2$$

$$= n_j \sigma_j^2 + \frac{n_j}{N^2} [\sum_{\ell=1}^{q} n_\ell (m_\ell - m_j)]^2$$

$$\therefore \quad N\sigma^2 = \sum_{K=1}^{q} \{n_k \sigma_k^2 + \frac{n_k}{N^2} [\sum_{\ell=1}^{q} n_\ell (m_\ell - m_j)]^2\}$$

$$= \sum_{k=1}^{q} n_k \sigma_k^2 + \frac{1}{N^2} \sum_{k=1}^{q} n_k \left[\sum_{\ell=1}^{q} n_\ell (m_\ell - m_k) \right]^2$$

$$= \sum_{k=1}^{q} n_k \sigma_k^2 + \frac{1}{N^2} \sum_{K=1}^{q} n_k \left(\sum_{\ell=1}^{q} n_\ell m_k \right)^2$$

But $\sum_{\ell=1}^{q} n_\ell m_\ell = NM_{ov}$

$$\therefore \quad N\sigma^2 = \sum_{k=1}^{q} n_k \sigma_k^2 + \frac{1}{N^2} \sum_{k=1}^{q} n_k \left[N^2 M_{ov}^2 - 2NM_{ov} \sum_{\ell=1}^{q} n_\ell m_k + \left(\sum_{\ell=1}^{q} n_\ell m_k \right)^2 \right]$$

$$= \sum_{K=1}^{q} n_k \sigma_k^2 + \frac{1}{N} \left[N^2 M_{ov}^2 - 2NM_{ov} \sum_{\ell=1}^{q} n_\ell m_k \sum_{\ell=1}^{q} n_\ell m_k \sum_{\ell=1}^{q} n_\ell m_k \right]$$

$$= \sum_{K=1}^{q} n_k \sigma_k^2 + NM_{ov}^2 - 2M_{ov} \sum_{\ell=1}^{q} n_\ell m_k + \frac{1}{N} \sum_{\ell=1}^{q} n_\ell m_k^2 \sum_{\ell=1}^{q} n_\ell$$

But $\sum_{\ell=1}^{q} n_\ell = \sum_{K=1}^{q} n_k = N$

$$\therefore \quad N\sigma^2 = \sum_{K=1}^{q} n_k \sigma_k^2 + NM_{ov}^2 - 2M_{ov} (NM_{ov}) + \sum_{\ell=1}^{q} n_\ell m_k^2$$

$$\therefore \quad N\sigma_{ov}^2 = \sum_{K=1}^{q} n_k \sigma_k^2 + \sum_{K=1}^{q} n_k m_k^2 - NM_{ov}^2 \qquad -(7)$$

If all the blocks of samples are the same size, i.e. all n_k are equal, then

$$N\sigma_{ov}^2 = \sum_{K=1}^{q} \sigma_k^2 + n \sum_{k=1}^{q} m_k^2 - NM_{ov}^2$$

And $N = qn$

$$\therefore \quad \sigma_{ov}^2 = \frac{\sum_{K=1}^{q} \sigma_k^2}{q} + \frac{\sum_{K=1}^{q} m_k^2}{q} - \frac{\left(\sum_{K=1}^{q} m_k\right)^2}{q^2} \qquad -(8)$$

APPENDIX B

ADDITION SUBROUTINES

Cycles	Label	Mnemonic	Comments	Bytes
			This routine can add numbers up to four bytes. Addend is placed in locations ADD to ADD+3, The augend is placed in AUG to AUG+3, Size specifies the number of bytes to be added. The X register is used for control.	
2	ADDITN:	CLC		1
2		LDX #00		2
4	LOOP:	LDA ADD, X	Zero page indexed	2
4		ADC AUG, X		2
4		STA AUG, X		2
2		INX		1
2		TXA	Can't use CPX because it affects the carry	1
3		EOR SIZE		2
2		BNE LOOP		2
6		RTS		1
			This routine requires 16 bytes ROM +9 bytes RAM. It takes 31 Microseconds for one byte and 21 microseconds for each additional byte.	

APPENDIX C

M-ATEC II SUBTRACT ROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	SUBTRT:		This routine handles numbers up to four bytes. The subtractor is placed in locations ADD to ADD+3. The subtrahend is placed in locations AUG to AUG+3. The result is placed in locations AUG to AUG+3. The size of the arguments is given in SIZE. The X register is used for control	
2		SEC	Set carry	1
2		LDX #00		2
4	LOOP:	LDA AUG, X	Load accumulator, zero page indexed, with first byte of the subtrahend	2
4		SBC ADD, X	Subtract first byte of subtractor	2
4		STA AUG, X	Store result in AUG	2
2		INX	Increment X register	1
2		TXA		1
3		EOR SIZE	Exclusive OR of X and SIZE	2
2		BNE LOOP	Result is zero if (X = SIZE)	2
6		RTS		1

SUBTRT requires 16 bytes ROM and the same bytes of RAM as the routine ADDITN. It takes 31μs for one byte numbers and 21μs for each additional byte.

APPENDIX D

MULTIPLICATION SUBROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
			This routine multiplies any combination of 1 and 2 byte numbers. The size of the multiplier is specified in MSIZE and the size of the result in SIZE. The multiplier is placed in locations MULP and MULP+1. The multiplicand goes into locations ADD and ADD+1. The result will be placed in AUG, AUG+1, AUG+2 and AUG+3. Subroutine ADDITN is called by this routine.	
2	MLTPLY:	TYA		1
3		PHA	Save Y register	1
2		LDA #01		2
3		CMP MSIZE	Check for size of multiplier	2
2		BEQ 1BYTE		2
2		LDY #16	*	2
2		LDA #00	* Initialize for	2
3		STA AUG+2	* two byte	2
3		STA AUG+3	* multiplier	2
3		JMP CONT1	*	3
2	1BYTE:	LDY #08	@	2
2		LDA #00	@ initialize for	2
3		STA AUG+1	@ one byte	2
3		STA AUG+2	@ multiplier	2
3		STA AUG+3	@	2
2	CONT1:	LDA #02		2
3		CMP MSIZE		2
2		CLC	clear carry	1
2		BNE CONT2		2
5		ROR MULP+1	If multiplier is two bytes then rotate right the M.S. byte	2

Cycles	Label	Mnemonic	Comments	Bytes
5	CONT2:	ROR MULP	Rotate right L.S. byte	2
2		BCC CONT3		2
6		JSR ADDITN	Add multiplicand to result if carry set	3
5	CONT3:	ROL ADD		2
5		ROL ADD+1		2
2		LDA #04		2
3		EOR SIZE	EOR not CMP because CMP affects the carry	2
2		BNE CONT4		2
5		ROL ADD+2	If four byte	2
5		ROL ADD+3	Result specified	2
2	CONT4:	DEY		1
2		BNE CONT1	Branch back for next shift	2
2		PLA		1
2		TAY	Restore Y register	1
6		RTS		1

This routine needs 65 bytes ROM
+same 9 bytes RAM as routine ADDITN
+3 bytes RAM extra.

A 1*1 byte takes a maximum of 748 microseconds
A 1*1 byte takes a minimum of 364 microseconds

A 2*2 byte takes a maximum time of 1188 microseconds
A 2*2 byte takes a minimum time of 530 microseconds

APPENDIX E

M-ATEC II DIVIDE ROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	DIVIDE:		This routine will divide either 2 or 3 byte numbers by 1 byte numbers. The dividend is placed in AUG to AUG+3 and the divisor is placed in ADD to ADD+3. The result will appear in MULP and MULP+2. The size of the dividend is given in MSIZE and SIZE and both X and Y registers are used for control.	
2		LDA #03		2
3		CMP MSIZE	Determine size of dividend	2
2		BEQ 3BYTE		2
2		LDA #24		2
3		STA SIZE	Load size with check count	2
2		LDA #00		2
3		STA MULP		2
3		STA MULP+1	Clear quotient buffers	2
2		LDX #16		2
2		LDX #01	Complete initialization for two byte dividend	2
3		JMP CONT1		3
2	3BYTE:	LDA #32	Start initialization for 3 byte	2
3		STA SIZE	dividend	2
2		LDA #00		2
3		STA MULP		2

Cycles	Label	Mnemonic	Comments	Bytes
3		STA MULP+1		2
3		STA MULP+2		2
2		LDX #02		2
2		LDY #24	Initialize shift count for 3 byte dividend	2
2	CONT1:	INY	Left justify divisor and	1
3		CPY SIZE	check for zero divisor.	2
2		BMI CONT2	If -ve divisor ≠ 0	2
2		DEY		1
2		SEC	Makes divisor appear as a 1 if it should be zero. This is to prevent crash of program	1
3		JMP CONT3		3
5	CONT2:	ASL ADD	Left shift divisor	2
2	CONT3:	BCC CONT1	If carry clear, divisor is not yet left justified.	2
5		ROR ADD	Bring most significant digit back into ADD from carry. Divisor is now left justified.	2
2		CLC		1
2	LOOP:	BCS CONT4	Branch if carry set	2
3		LDA AUG+2		2
3		CMP ADD		2
2		BMI CONT5		2
2		SEC		1
5		ROL MULP	Left shift quotient making lsb	2
5		ROL MULP+1	equal to 1	2

Cycles	Label	Mnemonic	Comments	Bytes
5		ROL MULP+2		2
2		SEC		1
4		LDA AUG, X		2
3		SBC ADD	Subtract divisor from most significant byte of dividend	2
4		STA AUG, X		2
3		JMP CONT6		3
2	CONT5:	CLC		1
5		ROL MULP		2
5		ROL MULP+1	Left shift quotient with l.s.b.	2
5		ROL MULP+2	becoming 0.	2
3		JMP CONT6		3
2	CONT4:	SEC		1
5		ROL MULP		2
5		ROL MULP+1	Left shift quotient with l.s.b.	2
5		ROL MULP+2	becoming 1	2
2		SEC		1
4		LDA AUG, X		2
3		SBC ADD		2
4		STA AUG, X		2
2	CONT6:	DEY		1
2		BEQ RETURN		2
2		LDA #03		2
3		CMP MSIZE		2

Cycles	Label	Mnemonic	Comments	Bytes
2		BEQ CONT7		2
2		CLC		1
5		ROL AUG	Shift two byte dividend	2
5		ROL AUG+1	to the left	2
3		JMP LOOP		3
2	CONT7:	CLC		1
5		ROL AUG	Shift three byte dividend	2
5		ROL AUG+1	to the left.	2
5		ROL AUG+2		2
3		JMP LOOP		3
6	RETURN:	RTS	Return from subroutine	1

Subroutine DIVIDE requires 131 bytes of ROM and uses 9 bytes of RAM in the arithmetic registers. The routine takes approximately 1000μs to execute.

APPENDIX F

M-ATEC II STATISTICS SUBROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	STATIS:		This subroutine handles all the statistics according to equations 2.1 to 2.4. Before the subroutine is called, the pointer is set to point to the current parameter. The latest value of the current parameter is in location TEMP1 when the subroutine is called. The Y register is used for indirect indexed addressing.	
3		LDA TEMP1	Recover sample value	2
3		STA ADD		2
2		LDA #02		2
3		STA SIZE	Specify precision of the addition	2
2		LDA #00		2
3		STA ADD+1	Clear ADD+1	2
3		STA AUG+2	Clear AUG+2	2
2		LDY #04	Y now points to CUMTOT	2
5		LDA (POINTL1),Y		2
3		STA AUG		2
2		INY		1
5		LDA (POINTL1),Y		2
3		STA AUG+1	Accumulated total is now in AUG	2
6		JSR ADDITN	Add new value to total	3
3		LDA AUG+1		2
6		STA (POINTL1),Y		2

Cycles	Label	Mnemonic	Comments	Bytes
2		DEY		1
3		LDA AUG		2
6		STA (POINTL1),Y	Replace new total	2
3		LDA ADD		2
3		STA MULP	Place sample value in multiplier register	2
2		LDA #00		2
3		STA MULP+1	Clear second byte of multiplier register	2
2		LDA #01		2
3		STA MSIZE	Set size of multiplier	2
6		JSR MLTPLY	Square the sample value Product is in AUG.	3
2		LDY #06	Set pointer to SQTOT	2
5		LDA (POINTL1),Y	;	2
3		STA ADD	;	2
2		INY	;	1
5		LDA (POINTL1),Y	; Place old square total in	2
3		STA ADD+1	; ADD register	2
2		INY	;	1
5		LDA (POINTL1),Y	;	2
3		STA ADD+2	;	2
2		LDA #03		2
3		STA SIZE	Set precision of the addition	2
6		JSR ADDITN	Add square of new value to total of squares	3

Cycles	Label	Mnemonic	Comments	Bytes
3		LDA AUG+2	;	2
6		STA (POINTL1),Y	;	2
2		DEY	;	1
3		LDA AUG+1	; Restore new square	2
6		STA (POINTL1),Y	; total	2
2		DEY	;	1
3		LDA AUG	;	2
6		STA (POINTL1),Y	;	2
2		LDA #K	K is no. of samples in a block	2
3		CMP SAMCNT	If K samples have not been	2
2		BNE RETURN	collected then return	2
2		LDX #R	$R = 2^K$	2
2		LDY #04	Set pointer to CUMTOT.	2
6		JSR MEAN	Calculate mean of the K samples, square it, and total both the mean and the mean squared	3
2		LDA #00		2
3		STA AUG	;	2
3		STA AUG+1	; Clear AUG register	2
3		STA AUG+2	;	2
2		LDX #R		2
2	Loop 3:	LDY #08	Set pointer to SQTOT+2	2
5		LDA (POINTL1),Y	;	2
2		LSR ACCUM	;	1

Cycles	Label	Mnemonic	Comments	Bytes
6		STA (POINTL1),Y	;	2
2		DEY	;	1
5		LDA (POINTL1),Y	;	2
2		ROR ACCUM	;	1
6		STA (POINTL1),Y	; This loop divides	2
2		DEY	; SQTOT by 2^R by	1
5		LDA (POINTL1),1	; shifting it to the	2
2		ROR ACCUM	; Right. The fractional part	1
6		STA (POINTL1),Y	; of the result appears in	2
2		ROR AUG	; AUG	1
2		DEX	;	1
2		BNE LOOP 3	;	2
3		STA AUG+1	AUG+1 ← SQTOT	2
2		INY	;	1
5		LDA (POINTL1),Y	; Move SQTOT+1 to AUG+2	2
3		STA AUG+2	; $\Sigma(x_i)^2/n$ is now in AUG	2
3		LDA TEMP1	;	2
3		STA ADD	; TEMP1 to TEMP3 were loaded with	2
3		LDA TEMP2	; the value of $(\Sigma x_i/n)^2$ in	2
3		STA ADD+1	; subroutine MEAN. This value	2
3		LDA TEMP3	; is placed in ADD register	2
3		STA ADD+2		2
6		JSR SUBTRT	Calculate variance. Result is placed in AUG & AUG+1	3

Cycles	Label	Mnemonic	Comments	Bytes
2		LDY #19	Set pointer to VARSUM	2
5		LDA (POINTL1),Y		2
3		STA ADD	Move VARSUM to ADD	2
2		INY		1
5		LDA (POINTL1),Y		2
3		STA (ADD+1)		2
2		LDA #02		2
3		STA SIZE		2
6		JSR ADDTN	Add new value to sum of variance	3
3		LDA AUG+1	;	2
6		STA (POINTL1),Y	;	2
2		DEY	; Replace new total of variances.	1
3		LDA AUG	;	2
6		STA (POINTL1),Y	;	2
2		LDA #32		2
3		CMP ANCTR	Check if current parameter is	2
2		BNE CLEAR	the last.	2
3		INC BLKCNT	If it is increment BLKCNT	2
2	CLEAR:	LDY #04	Set pointer to CUMTOT	2
2		LDA #00		2
6		STA (POINTL1),Y	;	2
2		INY	;	1
6		STA (POINTL1),Y	;	2

Cycles	Label	Mnemonic		Bytes
2		INY	; Clear CUMTOT and	1
6		STA (POINTL1),Y	; MNTOT ready	2
2		INY	; for collection of the next	1
6		STA (POINTL1),Y	; block of samples	2
2		INY	;	1
6		STA (POINTL1),Y	;	2
2		LDA #N	N is the number of blocks of samples which must be collected before next calculation of mean & variance is made	1
3		CMP BLKCNT	Check if N blocks collected	2
2		BNE RETURN	If not then return to ANVSCN	2
2		LDA #10		2
4		ORA PIA3A	Disable PATE interrupt signal	3
2		LDY #09	Set points to CUMTOT	2
2		LDX #H	where $2^H = N$	2
6		JSR MEAN	Calculate mean etc.	3
2		LDA #00		2
3		STA AUG		2
3		STA AUG+1		2
3		STA AUG+2		2
2		LDX #H		2
2	LOOP5:	LDY #20	;	2
5		LDA (POINTL1),Y	;	2
2		LSR ACCUM	;	1

Cycles	Label	Mnemonic	Comments	Bytes
5		STA (POINTL1),Y	;	2
2		DEY	; Divide varsum by 2H by	1
5		LDA (POINTL1),Y	; shifting to the right	2
2		ROR ACCUM	;	1
5		STA (POINTL1),Y	;	2
2		ROR AUG	;	1
2		DEX	;	1
2		BNE LOOP5	; Now $\sum_{1}^{N} \sigma^2/N$ is in AUG	2
5		LDA (POINTL1),Y	; register.	2
3		STA AUG+1	;	2
2		LDX #H		2
2	LOOP6:	LDY #13		2
5		LDA (POINTL1),Y	;	2
2		LSR ACCUM	;	1
6		STA (POINTL1),Y	;	2
2		DEY	;	1
5		LDA (POINTL1),Y	;	2
2		ROR ACCUM	; Divide SQMNSUM by N	1
6		STA (POINTL1),Y	; by shifting to the right	2
2		DEY	; Result is placed in ADD	1
5		LDA (POINTL1),Y	; It is $\sum m_N^2/N$ and is	2
2		ROR ACCUM	; in ADD & ADD+1	1
6		STA (POINTL1),Y	;	2
3		ROR ADD	;	2

Cycles	Label	Mnemonic	Comments	Bytes
2		DEX ;		1
2		BNE LOOP6 ;		2
5		LDA (POINTL1),Y ;		2
3		STA ADD+1		2
2		LDA #03		2
3		STA SIZE	Set size of addition	2
2		LDA #00		2
3		STA ADD+2		2
3		STA AUG+2		2
6		JSR ADDITN	This puts $\Sigma\ \sigma_N^2/N + \Sigma\ m_N^2/N$ in AUG register	3
3		LDA TEMP1	TEMP1, 2 and 3 were loaded in	2
3		STA ADD	subroutine mean with the value	2
3		LDA TEMP2	of $(\Sigma m_N/N)^2$	2
3		STA ADD+1		2
3		LDA TEMP3		2
3		STA ADD+2		2
6		JSR SUBTRT	Value of overall variance is now in AUG register	3
2		LDA #02		2
3		STA SIZE	Set size of next addition	2
2		LDY #21	Set pointer to OVVARSUM	2
5		LDA (POINTL1),Y		2
3		STA ADD		2

Cycles	Label	Mnemonic	Comments	Bytes
2		INY		1
5		LDA (POINTL1),Y		2
2		STA (ADD+1)		2
6		JSR ADDITN	Add new overall variance to total	3
2		LDA AUG+1	;	2
6		STA (POINTL1),Y	;	2
2		DEY	; Replace new value of	1
3		LDA AUG	; OVVARSUM	2
6		STA (POINTL1),Y	;	2
2		LDY #23		2
5		LDA (POINTL1),Y		2
2		INC ACCUM	Increment OVCNT	1
6		STA (POINTL1),Y		2
2		LDA #00		2
2		LDY #09		2
6		STA (POINTL1),Y	;	2
2		INY	;	1
6		STA (POINTL1),Y	;	2
2		INY	;	1
6		STA (POINTL1),Y	; Clear MNTOT	2
2		INY	; SQSUM and VARSUM	1
6		STA (POINTL1),Y	; ready for new values	2
2		INY	;	1

Cycles	Label	Mnemonic	Comments	Bytes
6		STA (POINTL1),Y ;		2
2		INY ;		1
6		STA (POINTL1),Y ;		2
2		INY ;		1
6		STA (POINTL1),Y ;		2
2		LDA #$F7		2
4		AND PIA3A	Clear bit 4 of PIA3A but leave other bits untouched. This re-enables the PATE interrupt	3
6		RTS		1

This routine requires 419 bytes of ROM and if only one group of statistics calculations is done, then with K = 128 and N = 256, the maximum execution time is 39.6ms. If both groups of calculations are done, then the maximum execution time is 97ms.

SYMBOL INTERPRETATION

ADD: An argument register for the math routines.

AUG: An argument register for the math routines.

BLKCNT: This records the number of blocks of K samples taken so far.

CUMTOT: This is the total of the sample values.

MNTOT: This is the sum of the means.

OVCNT: The number of groups of N*K samples is recorded in OVCNT.

OVVARSUM: This is the sum of the variances of each group of N*K samples.

SIZE: This is used to set the precision of an addition or subtraction.

SQMNSUM: This is the sum of the means of each block of K samples, squared.

SQTOT: This is the sum of the sample values squared.

TEMP1→3: These locations are used for saving intermediate values.

VARSUM: This is the sum of the variances of each block of K samples.

APPENDIX G

M-ATEC II MEAN SUBROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	MEAN:		This routine takes the accumulated total or the total of means and calculates the mean or the overall mean and adds the result to a total. It then squares the mean it has calculated and adds that to a total. Before MEAN is called the Y register is set to point to either CUMTOT or MNTOT and the X register with the power of two by which the total must be divided to obtain the mean.	
2		LDA #00		2
3		STA ADD	Clear ADD	2
2	LOOP1:	INY		1
5		LDA (POINTL1),Y	; This loop divides the location	2
2		LSR ACCUM	; pointed to by Y, (Either CUMTOT	1
6		STA (POINTL1),Y	; or MNTOT) by shifting	2
2		DEY	; them to the right the number of	1
5		LDA (POINTL1),Y	; times given in the	2
2		ROR ACCUM	; X register.	1
6		STA (POINTL1),Y	; The fractional part of the result	2
5		ROR ADD	; goes into location ADD	2
2		DEX	;	1
2		BNE LOOP1	;	2
3		STA ADD+1	place integer part of result	2
3		STA TEMP3	in ADD+1	2
3		LDA ADD		2

Cycles	Label	Mnemonic	Comments	Bytes
3		STA TEMP2	Store full result for future use	2
5		ASL ADD		2
2		BCC CONT1		2
5		INC ADD+1	Round up the result.	2
3	CONT1:	LDA ADD+1		2
3		STA ADD	Move ADD+1 to ADD	2
2		LDA #00		2
3		STA ADD+1	Clear ADD+1	2
2		TYA		1
2		ADC #04	Set Y to point to either MNTOT or OVMNTOT	2
2		TAY	Restore Y register	1
5		LDA (POINTL1),Y	;	2
3		STA AUG	;	2
2		INY	; Place total in	1
5		LDA (POINTL1),Y	; AUG	2
3		STA AUG+1	;	2
2		LDA #02		2
3		STA SIZE	Set size of arguments	2
6		JSR ADDITN	Add value of mean squared to total.	3
3		LDA AUG+1		2
5		STA (POINTL1),Y		2
2		DEY		1
3		LDA AUG		2

Cycles	Label	Mnemonic	Comments	Bytes
5		STA (POINTL1),Y	Replace new total	2
2		LDA #02		2
3		STA MSIZE	Set size of multiplier	2
2		LDA #04		2
3		STA SIZE		2
3		LDA TEMP2	;	2
3		STA ADD	;	2
3		STA MULP	; Set up registers for multiply	2
3		LDA TEMP3	; routine	2
3		STA ADD+1	;	2
3		STA MULP+1	;	2
6		JSR MLTPLY	Square the full value of the mean	3
5		ASL AUG		2
2		BCC CONT2		2
5		INC AUG+1	;	2
2		BNE CONT2	;	2
5		INC AUG+2	; Round up result to	2
2		BNE CONT2	; one byte of fractions	2
5		INC AUG+3	;	2
2	CONT2:	LDA #00		2
3		STA AUG	Clear least significant byte of fractions	2
3		LDA AUG+1	;	2
3		STA TEMP1	;	2

Cycles	Label	Mnemonic	Comments	Bytes
3		LDA AUG+2	; Save value of mean squared	2
3		STA TEMP2	; which has one byte of	2
3		LDA AUG+3	; fractions	2
3		STA TEMP3	;	2
5		ASL AUG+1	Now round to an integer	2
2		BCC CONT3		2
5		INC AUG+2	;	2
2		BNE CONT3	; Round up	2
5		INC AUG+3	;	2
3	CONT3:	LDA AUG+2		2
3		STA AUG	Shift AUG+2 to AUG	2
3		LDA AUG+3		2
3		STA AUG+1	Shift AUG+3 to AUG+1	2
2		LDA #00		2
3		STA AUG+2		2
2		INY		1
5		LDA (POINTL),Y	;	2
3		STA ADD	;	2
2		INY	;	1
5		LDA (POINTL1),Y	;	2
3		STA ADD+1	; Move total of means squared	2
2		INY	; into ADD	1
5		LDA (POINTL1),Y	;	2
3		STA ADD+2	;	2

Cycles	Label	Mnemonic	Comments	Bytes
2		LDA #03		2
3		STA SIZE	Set size of addition	2
6		JSR ADDITN	Add value of mean squared to total	3
3		LDA AUG+2	;	2
6		STA (POINTL1),Y	;	2
2		DEY	;	1
3		LDA AUG+1	; Replace new value of	2
6		STA (POINTL1),Y	; total in memory	2
2		DEY	;	1
3		LDA AUG	;	2
6		STA (POINTL1),Y	;	2
6		RTS	Return from subroutine	1

The subroutine requires 187 bytes ROM and uses the same RAM as the statistics routine. With K=128 the routine executes for a maximum time of 567μs and for a minimum time of 531μs.

SYMBOL INTERPRETATION

ACCUM: CPU accumulator.

ADD: Maths argument register.

AUG: Maths argument register.

CUMTOT: Total of sample values.

MNTOT: Total of means of each block of K samples.

MSIZE: This sets size of multiplier in a multiplication.

MULP: Math argument register.

OVMNTOT: Total of means of each group of N*K samples.

TEMP1→3: Used for saving intermediate results.

APPENDIX H

PATE INTERRUPT ROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	PATINT:		This routine services the interrupt which comes from the PATE once an hour. It returns the mean and variance for the preceding hour	
2		CLI	Enable interrupt	1
2		TXA		1
3		PHA	Save X register	1
2		TYA		1
3		PHA	Save Y register	1
3		LDA ADD		2
3		PHA	Save ADD	1
3		LDA ADD+1		1
3		PHA	Save ADD+1	1
3		LDA ADD +2		2
3		PHA	Save ADD+2	1
3		LDA AUG		2
3		PHA	Save AUG	1
3		LDA AUG+1		2
3		PHA	Save AUG+1	1
3		LDA AUG+2		2
3		PHA	Save AUG+2	1
3		LDA SIZE		2
3		PHA	Save SIZE	1

Cycles	Label	Mnemonic	Comments	Bytes
3		LDA MSIZE		2
3		PHA	Save MSIZE	1
3		LDA MULP		2
3		PHA	Save MULP	1
3		LDA MULP+1		2
3		PHA	Save MULP+1	1
3		LDA MULP+2		2
3		PHA	Save MULP+2	1
3		LDA TEMP1		2
3		PHA	Save TEMP1	1
3		LDA TEMP2		2
3		PHA	Save TEMP2	1
3		LDA TEMP3		2
3		PHA	Save TEMP3	1
2		LDA #32		2
3		STA CONCNT	Set up control counter	2
2		LDA #$02		2
3		STA POINTH3	Set up high pointer	2
2		LDA #$0E		2
3		STA POINTL3	Set up low pointer to point to OVMNTOT	2
2	LOOP1:	LDY #$07	Set Y register to point to OVVARSUM	2
5		LDA (POINTL3),Y		2
3		STA AUG	Move OVVARSUM to AUG	2

Cycles	Label	Mnemonic	Comments	Bytes
2		INY		1
5		LDA (POINTL3),Y		2
3		STA AUG+1		2
2		INY		1
5		LDA (POINTL3),Y		2
3		STA ADD	Move OVCNT to ADD	2
2		LDA #02		2
3		STA MSIZE	Specify No. of bytes in division.	2
6		JSR DIVIDE	Calculate OVVARSUM/OVCNT	3
3		LDA MULP		2
3		STA TEMP1		2
3		LDA MULP+1		2
3		STA TEMP2	Save result of calculation	2
2		LDY #$02		2
5		LDA (POINTL),Y		2
3		STA AUG	Move OVSQSUM to AUG	2
2		INY		1
5		LDA (POINTL3),Y		2
3		STA AUG+1	Move OVSQSUM+1 to AUG+1	2
2		INY		1
5		LDA (POINTL3),Y		2
3		STA AUG+2		2
2		LDY #09		2

Cycles	Label	Mnemonic	Comments	Bytes
5		LDA (POINTL3),Y		2
3		STA ADD	Move OVCNT to ADD	2
2		LDA #03		2
3		STA MSIZE	Specify no. of bytes	2
6		JSR DIVIDE	Calculate OVSQSUM/OVCNT	3
3		LDA MULP	;	2
3		STA ADD	;	2
3		LDA MULP+1	; Move result to ADD register	2
3		STA ADD+1	;	2
3		LDA MULP+2	;	2
3		STA ADD+2	;	2
3		LDA TEMP1	;	2
3		STA AUG	;	2
3		LDA TEMP2	; Move OVVARSUM/OVCNT to AUG	2
3		STA AUG+2	; register	2
2		LDA #03		2
3		STA SIZE		2
6		JSR ADD		3
3		LDA AUG		2
3		STA TEMP1		2
3		LDA AUG+1		2
3		STA TEMP2		2
3		LDA AUG+2		2
3		STA TEMP3		2

Cycles	Label	Mnemonic	Comments	Bytes
2		LDY #00		2
5		LDA (POINTL3),Y		2
3		STA AUG	Move OVMNTOT to AUG	2
2		INY		1
5		LDA (POINTL3),Y		2
3		STA AUG+1	Move OVMNTOT to AUG+1	2
2		LDA #00		2
3		STA AUG+2	Clear AUG+2	2
2		LDY #$09		2
5		LDA (POINTL3),Y		2
3		STA ADD		2
2		LDA #02		2
3		STA MSIZE	Specify no. of bytes	2
6		JSR DIVIDE	Calculate mean for one hour	3
3		LDA MULP+1		2
5		ROL MULP		2
2		BCC CONT1		2
2		CLC		1
2		ADC #01	Round up mean	2
3	CONT1:	ROR MULP	Restore MULP	2
2		LDY #$19	Set pointer to HRMEAN	2
6		STA (POINTL3),Y	Save value of mean in HRMEAN	2
3		LDA MULP	;	2
3		STA ADD	; Copy value of mean to ADD	2

Cycles	Label	Mnemonic	Comments	Bytes
3		LDA MULP+1	;	2
2		STA ADD+1	;	2
3		LDA #00	;	2
3		STA ADD+2		2
3		STA ADD+3		2
3		STA AUG		2
3		STA AUG+1		2
3		STA AUG+2		2
3		STA AUG+3		2
6		JSR MULT	Square full value of mean	3
5		ROL AUG		2
2		BCC CONT2		2
5		INC AUG+1		2
2		BNE CONT2		2
5		INC AUG+2		2
2		BNE CONT2		2
5		INC AUG+3		2
3	CONT2:	LDA AUG+1		2
3		STA AUG		2
3		LDA AUG+2		2
3		STA AUG+1		2
3		LDA AUG+3		2
3		STA AUG+2		2
3		LDA TEMP1		2

Cycles	Label	Mnemonic	Comment	Bytes
3		STA ADD		2
3		LDA TEMP2		2
3		STA ADD+1		2
3		LDA TEMP3		2
3		STA ADD+2		2
2		LDA #03		2
3		STA SIZE		2
6		JSR SUBTRT	Calculate variance	3
2		LDX #03		2
5	LOOP2:	ROL AUG		2
5		ROL AUG+1		2
2		DEX		1
2		BNE LOOP2		2
5		ROL AUG		2
2		BCC CONT3		2
5		INC AUG+1		2
3	CONT3:	LDA AUG+1	Variance packed as 5.3 bits is loaded into accum.	2
2		INY		1
6		STA (POINTL3),Y	Save variance in HRVRNCE	2
2		LDA #00		2
2		LDY #00		2
6		STA (POINTL3),Y	Clear OVMNTOT	2
2		INY		1

Cycles	Label	Mnemonic	Comment	Bytes
6		STA (POINTL3),Y	Clear OVMNTOT+1	2
2		INY		1
6		STA (POINTL3),Y	Clear OVSQSUM	2
2		INY		1
6		STA (POINTL3),Y	Clear OVSQSUM+1	2
2		INY		1
6		STA (POINTL3),Y	Clear OVSQSUM+6	2
2		LDY #$07		2
6		STA (POINTL3),Y	Clear OVVARSUM	2
2		INY		1
6		STA (POINTL3),Y	Clear OVVARSUM+1	2
2		INY		1
6		STA (POINTL3),Y	Clear OVCNT	2
2		LDA #$1A		2
2		CLC		1
3		ADC POINTL3		2
2		BCC BRNCH1		2
5		INC POINTH3	Set pointer for next variable	2
2	BRNCH1:	DEC CONCNT		1
2		BNE LOOP1		2
2		LDA #$07	If all 32 parameters dealt with	2
3		STA CONCNT	then set up control counter for TTL count mean calculations	2
2		LDA #$61		2
3		STA POINTL3		2

Cycles	Label	Mnemonic	Comments	Bytes
2		LDA #$05		2
3		STA POINTH3	Set up pointer to OVMNSUM	2
2		LDY #00	Set pointer to TTL CUMTOT	2
5	LOOP3:	LDA (POINTL3),Y		2
3		STA AUG	Move OVMNSUM to AUG	2
2		INY		1
5		LDA (POINTL3),Y		2
3		STA AUG+1	Move OVMNSUM+1 to AUG+1	2
2		INY		1
5		LDA (POINTL3),Y		2
3		STA ADD	Move OVCNT to ADD	2
2		LDA #02		2
3		STA MSIZE		2
6		JSR DIVIDE	Calculate mean of current counter value	3
5		ROL MULP		2
2		BCC CONT4		2
5		INC MULP+1		2
3	CONT4:	LDA MULP+1		2
2		LDY #$0D		2
6		STA (POINTL3),Y	Store mean in counter HRMEAN	2
2		LDY #$08		2
2		LDA #00		2
6		STA (POINTL3),Y		2

Cycles	Label	Mnemonic	Comments	Bytes
2		DEY		1
6		STA (POINTL3),Y		2
2		DEY		1
6		STA (POINTL3),Y	Clear OVMNSUM and OVCNT	2
2		LDA #$0E		2
2		CLC		1
3		ADC POINTL3	Set pointers to base for next counter	2
2		BCC BRNCH2		2
5		INC POINTH3		2
2	BRNCH2:	DEC CONCNT		1
2		BNE LOOP3		2
4		PLA		1
3		STA TEMP3		2
4		PLA		1
3		STA TEMP2		2
4		PLA		1
3		STA TEMP1		2
4		PLA		1
3		STA MULP+2		2
4		PLA		1
3		STA MULP+1		2
4		PLA		1
3		STA MULP		2

Cycles	Label	Mnemonic	Comments	Bytes
4		PLA		1
3		STA MSIZE		2
3		PLA		1
4		STA SIZE		2
3		PLA		1
3		STA AUG+2		2
4		PLA		1
3		STA AUG+1		2
4		PLA		1
3		STA AUG		2
4		PLA		1
3		STA ADD+2		2
4		PLA		1
3		STA ADD+1		2
4		PLA		1
3		STA ADD		2
4		PLA		1
2		TAY		1
4		PLA		1
2		TAX		1
4		PLA	Restore all registers	1
6		RTI	Return from interrupt	1

This routine requires 449 bytes of ROM and two bytes of RAM for a pointer and one for a control counter. It executes for a maximum time of 18.8ms.

SYMBOL INTERPRETATION

ADD:	Maths argument register.
AUG:	Maths argument register.
CONCNT:	Control counter; keeps track of current parameter.
HRMEAN:	Location for hourly value of mean.
HRVRNCE:	Location for hourly variance.
MSIZE:	Sets size of multiplier in multiplication or dividend in division.
MULP:	Maths argument register.
OVCNT:	Keeps count of number of groups of N*K samples for both analogue voltages & TTL counters.
OVMNSUM:	Sum of means of each group of N*K samples of TTL counters.
OVMNTOT:	Sum of means of each group of N*K samples of analogue voltages.
OVSQSUM:	Sum of means, squared, of each group of N*K samples of analogue voltages.
OVVARSUM:	Sum of variances of each groups of N*K samples of analogue voltages & TTL counts.
TEMP1→3:	Used for saving intermediate values.

APPENDIX I

ALARM LEVEL TEST SUBROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	ALMTST:		This subroutine compares the sample value of the current parameter with stored alarm levels. The subroutine is called from ANVSCN and the sample value is in location TEMP1. The program uses a pointer, set to the current parameter in ANVSCN, to locate the alarm level desired.	
2		LDY #00	Set pointer to RED LOW level	2
5		LDA (POINTL1),Y		2
3		CMP TEMP1		2
2		BMI AMBLOW	If -ve, branch and continue test	2
2		LDA #01	else signal red low alarm	2
3		STA TEMP2		2
3		JMP END	Jump to end of routine	3
2	AMBLOW:	INY	Set pointer to AMBER LOW level	1
5		LDA (POINTL1),Y		2
3		CMP TEMP1		
2		BMI REDHGH	If -ve branch and continue test	2
2		LDA #02	else signal amber low alarm	2
3		STA TEMP2		
3		JMP END		3
2	REDHGH:	INY	Set pointer to RED HIGH level	1
5		LDA (POINTL1),Y		2
3		CMP TEMP1		2

Cycles	Label	Mnemonic	Comments	Bytes
2		BPL AMBHGH	If +ve branch and continue test	2
2		LDA #03	else signal red high alarm	2
3		STA TEMP2		2
3		JMP END		3
2	AMBHGH:	INY	Set pointer to AMBER HIGH level	1
5		LDA (POINTL1),Y		2
3		CMP TEMP1		2
2		BPL GREEN	If +ve branch and signal green	2
2		LDA #04	else signal amber high	2
3		STA TEMP2	alarm	2
3		JMP END		3
2	GREEN:	LDA #00		2
3		STA TEMP2		2
2	END:	LDY #24	Set pointer to alarm state location	2
3		LDA TEMP2		2
6		STA (POINTL1),Y	Store alarm level in alarm state	2
6		RTS		1

This subroutine requires 68 bytes of ROM and uses TEMP1 and TEMP2. It also has 1 byte of RAM in the measured parameter list. If there is no alarm the routine executes for 72μs. If the red low alarm condition is signalled the routine executes for 37μs. Other alarm levels result in execution times between these levels.

APPENDIX J

ANALOGUE VOLTAGE SCAN ROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	ANVSCN:		This routine controls the analogue voltage scan interface reading in the samples of each of the 32 parameters and calling the statistics and alarm check subroutines as necessary. The program uses several control bytes. These are: BLKCNT, SAMCNT which keep track of the number of samples and blocks of samples read in. ANCTR which keeps track of the number of the current parameter. ANCSW which selects the input to the analogue multiplexer CR2B control register of PIA2B	
2		LDA #32		2
3		STA ANCTR	Load input port counter	2
2		LDA #01		2
3		STA ANCSW	Set control word	2
5		INC SAMCNT	Increment no. of samples count	2
2		LDA #00		2
3		STA POINTL1	Set low byte of pointer to first parameter	2
2		LDA #02		2
3		STA POINTH1	Set high byte of pointer to first parameter	2
3	LOOP1:	LDA ANCSW		2
4		STA PIA2A	Select desired parameter	3

Cycles	Label	Mnemonic	Comments	Bytes
2		LDA #32		2
3		CMP ANCSW	If this is parameter #32 (eye opening)	2
2		BEQ READ	then jump and read in value	2
2		LDA #$3D	else do A/D conversion	2
4		STA CR2B	Emit start conversion pulse	3
2		LDA #$35		2
4		STA CR2B	Clear start conversion pulse	3
4	LOOP2:	BIT CR2B	This causes bit 7 (msb) of CR2B to be loaded into the -ve (N) flag. This bit is set when conversion is is complete therefore branch and	
2		BPL LOOP2	check again if it is not set	2
2		LDA #$21	If conversion complete	2
4		STA PIA2A	then enable TSB for analogue multiplexer	3
4	READ:	LDA PIA2B	Read in value of sample	3
3		STA TEMP1	Store value in TEMP1	2
6		JSR ALMTST	Check to see if value is on alarm level	3
6		JSR STATIS	Calculate statistics	3
5		DEC ANCTR	If all 32 parameters have been sampled	2
2		BEQ EXIT	then exit from routine	2
5		INC ANCSW	else do, set control word to next parameter	
2		LDA #$1B		2
3		ADC POINTL1	; Add 28 to low byte of pointer so that	2

Cycles	Label	Mnemonic	Comments	Bytes
2		BCC JUMP	; it points to next parameter	2
5		INC POINTH1	;	2
3	JUMP:	JMP LOOP1		3
2	EXIT:	LDA #K		2
3		CMP SAMCNT		2
2		BNE CLRBLK		2
2		LDA #00		2
3		STA SAMCNT	Clear sample count if it is K	2
2	CLRBLK:	LDA #N		2
3		CMP BLKCNT		2
2		BNE RETURN		2
2		LDA #00		2
3		STA BLKCNT	Clear block count if it is N	2
6	RETURN:	RTS	Return from subroutine	1

This routine requires 96 bytes of ROM. It executes for a maximum of 36.6 ms if no statistics calculations are done. This includes time for the A/D conversion. If the first group of statistics calculations is done then, with K = 128 and N = 256 the maximum execution time is 76.2 ms. If both groups are done then the time taken is 133.6 ms.

APPENDIX K

TTL INTERRUPT ROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
			FTTL/MTTL SERVICE ROUTINE Use locations POINTL2, POINTH2 as pointer to constants for next count to be read as with ANVSCN. These are incremented as each interrupt is serviced to point to the next one.	
2	TTLINT:	SEI	Disable IRQ	1
3		PHA	Save accumulator	1
2		TXA		1
3		PHA	Save X register	1
2		TYA		1
3		PHA	Save Y register	1
3		LDA ADD		2
3		PHA	Save ADD	1
3		LDA ADD+1		2
3		PHA	Save ADD+1	1
3		LDA AUG		2
3		PHA	Save AUG	1
3		LDA AUG+1		2
3		PHA	Save AUG+1	1
2		LDA #$B0		2
4		STA PIA1A	Reload TBG no. 1	3
3		LDA TSBN		2
4		STA PIA2A	Enable TSB given by TSBN	3
4		LDA PIA2B	Read in count	3
3		STA ADD		2
3		LDA CNTN		2
4		STA PIA2A	Clear counter given by CNTN	3

Cycles	Label	Mnemonic	Comments	Bytes
2		LDY #$00		2
2		LDA #$00		2
3		STA ADD+1	Clear ADD+1	2
5		LDA (POINTL2),Y	Load accum. with CUMTOT	2
3		STA AUG		2
2		INY		1
5		LDA (POINTL2),Y		2
3		STA AUG+1	Load CUMTOT1 into AUG+1	2
2		LDA #02		2
3		STA SIZE	Choose precision	2
6		JSR ADD		3
2		INY		1
5		LDA (POINTL2),Y		2
2		CLC	Clear carry	1
2		ADC #01	Inc. no. of samples count (SAMCNT)	2
2		CMP #K	No. of samples = K?	2
2		BEQ CONT1		2
6		STA (POINTL2),Y	Restore SAMCNT	2
2		DEY		1
3		LDA AUG+1		2
6		STA (POINTL2),Y	Restore CUMTOT1	2
2		DEY		1
3		LDA AUG		2
6		STA (POINTL2),Y	Restore CUMTOT	2
3		JMP SETPNT	Jump to set pointers for next call of this routine	3
2	CONT1:	LDX #J	Where 2**J = K	2
2	LOOP1:	CLC	Clear carry	1
5		ROR AUG+1		2
5		ROR AUG		2

Cycles	Label	Mnemonic	Comments	Bytes
2		DEX		1
2		BNE LOOP1	Divide by K	2
2		BCC RNDDWN		2
5		INC AUG	Round up	2
2	RNDDWN:	INY		1
5		LDA (POINTL1),Y		2
3		STA ADD	Load MNSUM into ADD	2
2		INY		1
5		LDA (POINTL1),Y		2
3		STA ADD+1	Load MNSUM+1 into ADD+1	2
6		JSR ADD	Add latest value of mean to total of means (MNSUM)	3
2		LDY #00		2
2		LDA #00		2
6		STA (POINTL2),Y	Clear CUMTOT	2
2		INY		1
6		STA (POINTL2),Y	Clear CUMTOT+1	2
2		INY		1
6		STA (POINTL2),Y	Clear SAMCNT	2
2		LDY #$05		2
5		LDA (POINTL2),Y	Load accum. with no. of blocks (BLKCNT)	2
2		CLC	Clear carry	1
2		ADA #01	Inc. accumulator	2
2		CMP #N	Have N blocks of samples been collected?	2
2		BEQ CONT2		2
6		STA (POINTL1),Y	Restore BLKCNT	2
2		DEY		1
3		LDA AUG+1		2
6		STA (POINTL1),Y	Restore MNSUM+1	2
2		DEY		1

Cycles	Label	Mnemonic	Comments	Bytes
3		LDA AUG		2
6		STA (POINTL2),Y	Restore MNSUM	2
3		JMP SETPNT	Jump to set pointers for next call of this routine	3
2	CONT2:	LDX #M	Where 2**M = N	2
2	LOOP2:	CLC	Clear carry	1
5		ROR AUG+1		2
5		ROR AUG		2
2		DEX		1
2		BNE LOOP2	Divide by N	2
2		BCC RNDDWN		2
5		INC AUG	Round up overall mean	2
2		INY		1
5		LDA (POINTL2),Y		2
3		STA ADD	Move OVMNSUM to ADD	2
2		INY		1
5		LDA (POINTL2),Y		2
3		STA ADD+1	Move OVMNSUM+1 to ADD+1	2
6		JSR ADD	Add new overall mean to total	3
2		INY		1
5		LDA (POINTL2),Y		2
2		ADC #01	Inc. no. of overall values count (OVCNT)	2
6		STA (POINTL2),Y		2
2		DEY		1

Cycles	Label	Mnemonic	Comments	Bytes
3		LDA AUG+1		2
6		STA (POINTL2),Y	Replace new value of OVMNSUM+1	2
2		DEY		1
3		LDA AUG		2
6		STA (POINTL2),Y	Replace new value of OVMNSUM	2
2		DEY		1
2		LDA #00		2
6		STA (POINTL2),Y	Clear BLKCNT	2
2		DEY		1
6		STA (POINTL2),Y	Clear MNSUM+1	2
2		DEY		1
6		STA (POINTL2),Y	Clear MNSUM	2
2	SETPNT:	LDY #$09		2
5		LDA (POINTL2),Y		2
3		STA TSBN	Set TSB no. for next call of this routine	2
5		INY		1
3		LDA (POINTL2),Y		2
2		STA CNTN	Set TTL counter no. for next call of this routine	2
5		INY		1
5		LDA (POINTL2),Y		2
3		STA TEMP 4	Fetch new low pointer value and store in TEMP 4	2
2		INY		1

Cycles	Label	Mnemonic	Comments	Bytes
5		LDA (POINTL2),Y		2
3		STA POINTH2	Set new high pointer	2
3		LDA TEMP4		2
3		STA POINTL2	Set new low pointer	2
4		PLA		1
3		STA AUG+1	Restore AUG+1	2
4		PLA		1
3		STA AUG	Restore AUG	2
4		PLA		1
3		STA ADD+1	Restore ADD+1	2
4		PLA		1
3		STA ADD	Restore ADD	2
4		PLA		1
2		TAY	Restore Y register	1
4		PLA		1
2		TAX	Restore X register	1
4		PLA	Restore accumulator	1
2		CLI	Enable IRQ	1
6		RTI	Return from interrupt	1

This routine requires 250 bytes ROM and 9 bytes RAM including 5 bytes on the stack. It also uses the registers for the arithmetic subroutine.

The routine usually executes in 304 μs. If the mean is calculated it takes 505 μs. If it calculates mean and overall mean it takes 566 μs.

SYMBOL INTERPRETATION

ADD: Maths argument registers.

AUG: Maths argument registers.

CNTN: Keeps the number of the current counter.

CUMTOT: Total of the sample values of the TTL counters.

MNSUM: Total of means of each block of K samples.

NCNTN: Keeps the number of the counter for the next call of this routine.

NTSBN: Keeps the number of the TSB for the next call of this routine.

OVCNT: Keeps the number of groups of N*K samples.

OVMNSUM: Keeps the sum of the means of each group of N*K samples.

SAMCNT: Keeps the number of samples taken.

SIZE: Sets size of addition or subtraction.

TEMP4: Used for storage of intermediate results.

TSBN: Keeps number of the TSB for the current counter.

APPENDIX L

ALARM SCAN PROGRAM

Cycles	Label	Mnemonic	Comments	Bytes
			This program reads in the five eight-bit alarm words and reformats them into eight five-bit alarm words then checks for major alarms and builds the data characters for transmission.	
2	CCRTN:	LDA #$08		2
4		STA PIA1A	Enable TSB no. 1	3
4		LDA PIA1B	Read alarm word (a)	3
6		INC PIA1A	Clear flip-flop 1	3
6		INC PIA1A	Enable TSB no. 2	3
3		STA TEMP1	Temporarily store word (a)	2
2		LDA #$10	This sets bit 1 which is the NACK bit of the begin character	2
3		STA NACKFLAG	Set non-acknowledge flag	2
3		LDA TEMP1		2
4		BIT CONTROL1	CONTROL1 contains 01	3
2		BNE CONT1	Branches if alarm not acknowledged	2
2		LDA #$00		2
3		STA NACKFLAG	Clears non-acknowledge flag	2
5	CONT1:	LSR TEMP1	Drop acknowledge bit from alarm word	2
2		LDA #$1F		2
3		AND TEMP1	accumulator now has 5 least significant alarm bits from TEMP1	2
2		BEQ STORE1	branches if no alarms	2

Cycles	Label	Mnemonic	Comments	Bytes
2		LDX #01		2
6		JSR MJACHK	Calls major alarm check sub-routine	3
3	STORE1:	STA CCLCN1	Store alarm word no. 1	2
2		LDA #$06		2
3		AND TEMP1	The 5 alarms of word 1 are masked out	2
2		LSR ACCUM		1
2		LSR ACCUM		1
2		LSR ACCUM		1
2		LSR ACCUM		1
2		LSR ACCUM	Last two alarms of word (a) are right justified in accum.	1
3		STA TEMP2		2
4		LDA PIA1B	Read alarm word (b)	3
6		INC PIA1A	Clear F/F2	3
6		INC PIA1A	Enable TSB no.3	3
3		STA TEMP1		2
2		LDA #$03		2
3		AND TEMP1	Accum. holds first 3 alarms of word (b)	2
2		ASL ACCUM		1
3		ORA TEMP2	Accum. holds first 3 alarms of word (b) and last 2 word (a)	2
2		BEQ STORE2		2
2		LDX #$02		2

Cycles	Label	Mnemonic	Comments	Bytes
6		JSR MJACHK	Check for major alarms	3
3	STORE2:	STA CCLCN2	Store alarm word 2	2
2		LDA #$F8		2
3		AND TEMP1	Accum. holds remaining 5 alarms of word (b)	2
2		LSR ACCUM		1
2		LSR ACCUM		1
2		LSR ACCUM	Right justify these alarms	1
2		BEQ STORE3		2
2		LDX #$03		2
6		JSR MJACHK		3
3	STORE3:	STA CCLCN3	Store alarm word 3	2
4		LDA PIA1B	Read alarm word (c)	3
3		STA TEMP1		2
2		LDA #$1F		2
3		AND TEMP1	Accum. holds first 5 alarms of word (c), right justified	2
2		BEQ STORE4		2
2		LDX #$04		2
6		JSR MJACHK		3
3	STORE4:	STA CCLCN4	Store alarm word 4	2
2		LDA #$E0		2
3		AND TEMP1	Accum. holds last 3 alarms of word (c), left justified	2
2		ROL ACCUM		1

Cycles	Label	Mnemonic	Comments	Bytes
2		ROL ACCUM		1
2		ROL ACCUM		1
2		ROL ACCUM	Remaining word (c) alarms now right justified.	1
2		CLC	Clear carry	1
3		STA TEMP2	Store remaining alarms of word (c) pending reading of next word	2
6		INC PIA1A	Enable TSB no. 4	3
4		LDA PIA1B	Read alarm word (d)	3
3		STA TEMP1		2
2		LDA #$03		2
3		AND TEMP1	Accum. holds first 2 alarms of word (d)	2
2		LSR ACCUM		1
2		LSR ACCUM		1
2		LSR ACCUM		1
3		ORA TEMP2	Accum. holds first 2 alarms of word (d) and last 3 of word (c)	2
2		BEQ STORE5		2
2		LDX #$05		2
6		JSR MJACHK		3
3	STORE5:	STA CCLCN5	Store alarm word five	2
2		LDA #$7C		2
3		AND TEMP1	Accum. holds middle 5 alarms of word (d)	2
2		ASR ACCUM		1

Cycles	Label	Mnemonic	Comments	Bytes
2		ASR ACCUM	Right justify the accum.	1
2		BEQ STORE6		2
2		LDX #$06		2
6		JSR MJACHK		3
3	STORE6:	STA CCLCN6	Store alarm word 6	2
2		LDA #$80		2
3		AND TEMP1	Accum. holds remaining alarm of word (d)	2
2		ROL ACCUM		1
2		ROL ACCUM	Right justify accum.	1
3		STA TEMP2		2
2		CLC	Clear carry	1
6		INC PIA1A	Enable TSB no.5	3
4		LDA PIA1B	Read alarm word (e)	3
3		STA TEMP1		2
2		LDA #$OF		2
3		AND TEMP1		2
2		ASL ACCUM	Accum. holds first four alarms of word (e) with a 0 to the right	1
3		ORA TEMP2	Accum. holds first four alarms of word (e) and final alarm of word (d)	2
2		BEQ STORE7		2
2		LDX #$07		2
6		JSR MJACHK		3

AD-A055 864 PURDUE UNIV LAFAYETTE IND SCHOOL OF ELECTRICAL ENGI--ETC F/G 17/2
A MICROPROCESSOR BASED PERFORMANCE MONITOR FOR COMMUNICATION NE--ETC(U)
MAY 78 M S CALLOW, S C CRIST, B A BLACK F30602-75-C-0082
UNCLASSIFIED RADC-TR-78-112 NL

3 OF 3
AD
A055864

END
DATE
FILMED
9-78
DDC

Cycles	Label	Mnemonic	Comments	Bytes
3	STORE7:	STA CCLCN7	Store alarm word 7	2
2		LDA #$FO		2
3		AND TEMP1	Accum. holds last 4 alarms of word (e)	2
2		LSR ACCUM		1
2		LSR ACCUM		1
2		LSR ACCUM		1
2		LSR ACCUM	Right justify last alarms	1
2		BEQ STORE8		2
2		LDX #$08		2
6		JSR MJACHK		3
3	STORE8	STA CCLCN8	Store alarm word 8	2
3		JMP ANVSCN	Jump to analogue voltage scan routine	3

This routine requires 219 bytes ROM and 11 bytes RAM.

The routine executes for 267 cycles if there are no alarms and for 496 cycles if every alarm occurs.

SYMBOL INTERPRETATION

ACCUM: CPU accumulator.

CCLCN: Locations where the alarm words are stored.

CONTROL1: ROM location containing 01 for use with a BIT instruction.

NACKFLAG: This flag is set if an alarm has not been acknowledged.

TEMP1-2: Used for intermediate storage.

APPENDIX M

MAJOR ALARM CHECK SUBROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	MJACHK:		This subroutine is passed an alarm word in the accumulator and the number of that alarm word is the X register. The alarm word passed in is checked to see if any of the alarms set are major.	
3		STA TEMP3	Save accumulator during subroutine	2
4		AND CHKWDO, X	AND accumulator with stored constant indicating major alarm bits for word no. given by X. (Zero page indexed addressing)	3
2		BEQ ANYALM	Branch to ANYALM if no major alarm is set.	2
2		LDA #$012		2
3		ORA AFLAG	; Set major alarm bit in	2
3		STA AFLAG	; AFLAG without altering previously set bits.	2
2		LDA #$20	Set bit 5 of accumulator	2
3		JMP CONT		3
2	ANYALM:	LDA #$04		2
3		ORA AFLAG	; Set any alarm bit in	2
3		STA AFLAG	; AFLAG	
2		LDA #40	Set bit 6 of accumulator	2
3	CONT:	ORA TEMP3	; this places the bit set in the	2
			; ACCUM along with the alarm bits.	
			; This becomes the MA or AA bit of	
			; the data character	
6		RTS		1
			This subroutine requires 29 bytes ROM and it executes for a maximum time of 28 μs.	

Symbol Interpretation

AFLAG: This is used to remember the occurrence of major or any alarms until the begin character is transmitted.

CHKWDO: This is the base address of the check words used to remember which alarms are major.

TEMP3: This is temporary storage location #3.

APPENDIX N

TBG2 INTERRUPT SERVICE ROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	CHRINT:		This routine is executed on receipt of the interrupt from TGB2. It keeps track of the character which is to be transmitted next and loads this character in the ACIA output register and programs the ACIA to transmit this character. If this character is the begin character, then this routine will build it.	
2		LDA #$10		2
4		STA PIA3A	Clear TBG2	3
2		TXA		1
3		PHA	Save X register	1
2		LDA #$54		2
4		STA CRAC	Program ACIA to transmit character	3
3		LDA TXPNT	Load accumulator with pointer to next word to be transmitted	2
2		BNE CONT	If TXPNT = 0 then assemble begin character else branch to load character into ACIA	2
2		LDA #$AO	Set 101 in bits 5-7 of begin character	2
3		ORA NACKFLAG	place NACK bit in begin character	2
3		ORA AFLAG	place MA and AA bits in begin character	2
3		STA CCLTNO	Store begin charcter	2
2		LDA #00		2
3		STA NACKFLAG	Clear NACKFLAG	2
3		STA AFLAG	Clear AFLAG	2
4	CONT:	BIT CONTROL2		3
2		TAX		3
2		BNE RESET	If TXPNT = 8 then branch to reset it	2
5		INC TXPNT	else increment TXPNT	2
3		JMP TXMIT		3

Cycles	Label	Mnemonic	Comments	Bytes
2	RESET:	LDA #00		3
3		STA TXPNT	Reset TXPNT	2
4	TXMIT:	LDA CCLTNO, X	Load character to be transmitted	3
4		STA OACIA	Store in ACIA output register	3
4		PLA		1
2		TAX	Restore X register	1
4		PLA	Restore accumulator	1
6		RTI		1

This routine requires 55 bytes of ROM and executes for a maximum time of 83 µs.

Symbol Interpretation

AFLAG: This byte contains the major alarm (MA) and any alarm (AA) bits of the begin character.

CONTROL2: This byte contains a constant value of $08 for use with BIT instructions.

CRAC: This is the ACIA control register.

NACKFLAG: This contains the NACK bit of the begin character.

OACIA: The ACIA output register.

TAPNT: A pointer to the next character to be transmitted.

APPENDIX O

DATA REQUEST SERVICE ROUTINE

Cycles	Label	Mnemonic	Comments	Bytes
	ACAINT:		When the ACIA has transmitted the data in its output register, it generates on interrupt, requesting more data. This routine services that interrupt by programming the ACIA to transmit blanks.	
2		LDA #$74	Set up control word.	2
4		STA CRAC	Store control word in ACIA	3
4		PLA	Restore accumulator	1
6		RTI	Return from interrupt	1
			This routine requires 7 bytes and executes for 16 µs.	

Symbol Interpretation

CRAC: This is the ACIA control register.

APPENDIX P

EYE MONITOR PROGRAM

Cycles	Label	Mnemonic	Comments	Bytes
	EYEMON:		This program controls the AGC and reference voltages for the eye pattern monitor.	
2		LDA #AGCINIT		2
3		STA PA20	Set initial AGC voltage	2
2		LDA #BINIT		2
3		STA PB20	Set initial reference voltage	2
3		STA PB30	Set dispersion output value	2
2	READ:	LDA #$80		2
3		STA PA30	Enable latch	2
2		LDA #$00		2
3		STA PA30	Disable latch	2
3		LDA PA30	Read sample value	2
2		CMP #$09	If sample is zero level	2
2		BMI READ	Then read another sample	2
2		CMP #$70	If sample < - 2d	2
2		BEQ DECRSE	Then branch to decrease gain	2
2		CMP #$ØF	If sample > 2d	2
2		BEQ DECRSE	Then branch to decrease gain	2
5		INC INCGN	Else increment increase gain counter	2
2		CMP #$0E	If sample > 2d-b	2
2		BEQ INCRSE	Then branch to increase b	2
2		CMP #$3Ø	If sample < - (2d-b)	2
2		BEQ INCRSE	Then branch to increase b	2
5		INC DECB	Else increment decrease b counter	2
2		LDA #03		2
3		CMP DECB	If decrease b counter ≠ 3	2
2		BNE GAIN	Then branch to check AGC	2
5		DEC PB20	Else decrement b reference	2
5		DEC PB30	decrement eye dispersion	2

Cycles	Label	Mnemonic	Comments	Bytes
3		JMP GAIN	jump to check AGC	3
5	INCRSE:	INC INCB	Increment increase b counter	2
2		LDA #09		2
3		CMP INCB	If increase b counter ≠ 9	2
2		BNE GAIN	Then branch to check AGC	2
5		INC PB20	Else increment b reference	2
5		INC PB30	increment eye dispersion	2
2	GAIN:	LDA #08		2
3		CMP INCGN	If increase gain counter ≠ 8	2
2		BNE READ	Then read in another sample	2
5		INC PA20	Else increment AGC voltage	2
3		JMP READ	read in another sample	3
5	DECRSE:	INC DECGN	Increment decrease gain counter	2
2		LDA #08		2
3		CMP DECGN	If decrease gain counter ≠ 8	2
2		BNE READ	Then read another sample	2
5		DEC PA20	Else decrement AGC voltage	2
3		JMP READ	read in another sample	3

The program requires 91 bytes of ROM and 4 bytes RAM. If sample is zero level execution time is 17 μs. If sample is > 2d or < -2d then execution time is 36 to 45 μs. If sample is > 2d-b or < - (2d-b) then execution time is 51 to 73 μs and if sample is < 2d-b or > - (2d-b) then execution time is 57 to 78 μs.

SYMBOL INTERPRETATION

DECB: Counter for decrease reference voltage events

DECGN: Counter for decrease gain events

INCB: Counter for increase reference voltage events

INCGN: Counter for increase gain events

PA20: A port of 6520. (Used for AGC)

PA30: A port of 6530. (Used to read samples)

PB20: B port of 6520. (Used for reference voltage)

PB30: B port of 6530. (Used for eye dispersion output)

APPENDIX Q

INTERRUPT REQUEST SERVICE PROGRAM

Cycles	Label	Mnemonic	Comments	Bytes
	IRQ:		This program is executed when an IRQ interrupt is serviced. It reads the priority encoder output and uses this to jump to the required interrupt routine after clearing the flip-flop for that interrupt.	
3		PHA	Save accumulator	1
4		LDA P1A3B	Read priority encoder output	3
2		AND #$07	Mask out those pins which are outputs	2
2		BEQ CHR	If output is 0 then branch to CHR	2
2		CMP #02	If output is 2	2
2		BEQ ACA	Then branch to ACA	2
2		LDA #$20	Else	2
4		STA P1A3B	Clear PATINT flip-flop	3
3		JMP PATINT	Jump to PATINT	3
2	CHR:	LDA #$80		2
4		STA P1A3B	Clear CHRINT flip-flop	3
3		JMP CHRINT	Jump to CHRINT	3
2	ACA:	LDA #$40		2
4		STA P1A3B	Clear ACAINT flip-flop	3
3		JMP ACAINT	Jump to ACAINT	3
			This program requires 36 bytes ROM and executes for 20 μs if CHRINT and 24 μs if any other.	

SYMBOL INTERPRETATION

P1A3B: The B part of P1A3. Used to interface the interrupt priority circuit to the CPU.

MISSION
of
Rome Air Development Center

RADC plans and conducts research, exploratory and advanced development programs in command, control, and communications (C^3) activities, and in the C^3 areas of information sciences and intelligence. The principal technical mission areas are communications, electromagnetic guidance and control, surveillance of ground and aerospace objects, intelligence data collection and handling, information system technology, ionospheric propagation, solid state sciences, microwave physics and electronic reliability, maintainability and compatibility.